Improving the Utility of Social Media with Natural Language Processing

A thesis presented

by

Bo Han

to

The Department of Computing and Information Systems

in total fulfillment of the requirements

for the degree of

Doctor of Philosophy

The University of Melbourne

Melbourne, Australia

February 2014

Produced on archival quality paper

Declaration

This is to certify that:

(i) the thesis comprises only my original work towards the PhD except where indicated in the Preface;

(ii) due acknowledgement has been made in the text to all other material used;

(iii) the thesis is fewer than 100,000 words in length, exclusive of tables, maps, bibliographies and appendices.

Signed: _____ Date: _____

Thesis advisor(s)
Timothy Baldwin
Paul Cook

Author
Bo Han

Improving the Utility of Social Media with Natural Language Processing

Abstract

Social media has been an attractive target for many natural language processing (NLP) tasks and applications in recent years. However, the unprecedented volume of data and the non-standard language register cause problems for off-the-shelf NLP tools. This thesis investigates the broad question of how NLP-based text processing can improve the utility (i.e., the effectiveness and efficiency) of social media data. In particular, text normalisation and geolocation prediction are closely examined in the context of Twitter text processing.

Text normalisation is the task of restoring non-standard words to their standard forms. For instance, *earthquick* and *2morrw* should be transformed into "earthquake" and "tomorrow", respectively. Non-standard words often cause problems for existing tools trained on edited text sources such as newswire text. By applying text normalisation to reduce unknown non-standard words, the accuracy of NLP tools and downstream applications is expected to increase. In this thesis, I explore and develop lexical normalisation methods for Twitter text. I shift the focus of text normalisation from a cascaded token-based approach to a type-based approach using a combined lexicon, based on the analysis of existing and developed text normalisation methods. The type-based method achieved the state-of-the-art end-to-end normalisation accuracy at the time of publication, i.e., 0.847 precision and 0.630 recall on a benchmark dataset. Furthermore, it is simple, lightweight and easily integrable which is particularly well suited to large-scale data processing. Additionally, the effectiveness of the proposed normalisation method is shown in non-English text normalisation and other NLP tasks and applications.

Geolocation prediction estimates a user's primary location based on the text of their posts. It enables location-based data partitioning, which is crucial to a range of tasks and applications such as local event detection. The partitioned location data can improve both the efficiency and the effectiveness of NLP tools and applications. In this thesis, I identify and explore several factors that affect the accuracy of text-based geolocation prediction in a unified framework. In particular, an extensive range of feature selection methods is compared to determine the optimised feature set for the geolocation prediction model. The results suggest feature selection is an effective method for improving the prediction accuracy regardless of geolocation model and location partitioning. Additionally, I examine the influence of other factors includ-

ing non-geotagged data, user metadata, tweeting language, temporal influence, user geolocatability, and geolocation prediction confidence. The proposed stacking-based prediction model achieved 40.6% city-level accuracy and 40km median error distance for English Twitter users on a recent benchmark dataset. These investigations provide practical insights into the design of a text-based normalisation system, as well as the basis for further research on this task.

Overall, the exploration of these two text processing tasks enhances the utility of social media data for relevant NLP tasks and downstream applications. The developed method and experimental results have immediate impact on future social media research.

Contents

List of Figures

List of Tables

Citations to Previously Published Work

Large portions of Chapter 3 have appeared in the following papers:

> Bo Han and Timothy Baldwin (2011), Lexical normalisation of short text messages: Makn sens a #twitter. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-HLT 2011)*, pages 368–378, Portland, USA, 2011.
>
> Bo Han, Paul Cook and Timothy Baldwin (2012), Automatically constructing a normalisation dictionary for microblogs. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL 2012)*, pages 421–432, Jeju, Republic of Korea, 2012.
>
> Bo Han, Paul Cook and Timothy Baldwin (2013), Lexical Normalisation of Short Text Messages. In *ACM Transactions on Intelligent Systems and Technology (TIST 2013)*, 4(1), pages 5:1–27, DOI=10.1145/2414425.2414430
>
> Bo Han, Paul Cook and Timothy Baldwin (2013), unimelb: Spanish Text Normalisation. In *Proceedings of the Tweet Normalization Workshop at SEPLN 2013 (Tweet-norm)*, pages 67–71, Madrid, Spain.

Large portions of Chapter 4 have appeared in the following papers:

> Bo Han, Paul Cook and Timothy Baldwin (2012), Geolocation Prediction in Social Media Data by Finding Location Indicative Words. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012)*, pages 1045–1062, Mumbai, India, 2012.
>
> Bo Han, Paul Cook and Timothy Baldwin (2013), A Stacking-based Approach to Twitter User Geolocation Prediction. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL 2013)*, Demo Session, pages 7–12, Sofia, Bulgaria.
>
> Bo Han, Paul Cook and Timothy Baldwin (2014), Text-based Twitter User Geolocation Prediction. In *Journal of Artificial Intelligence Research*, pages 49:451–500.

Acknowledgments

First, I would like to express my heartfelt gratitude to Tim Baldwin and Paul Cook. As supervisors, they worked very hard with me. From discussions on concrete research topics to low level feedback on my writing samples, they were extremely accessible and supportive. Tim has a wide range of interests as a NLP researcher, and he is also an experienced educator. He encouraged me to pursue my ideas and also tolerated my sometimes naive questions. Whenever I felt lost in details, Tim was always there to help me out of the quagmire. Working with Paul was fruitful and enjoyable. From developing arguments in paper writing, to disseminating my research to the public, Paul kindly handed on his experience and skills, which not only added great value to my PhD, but has benefited my future career. I am also grateful to Steven Bird and Justin Zobel for kindly being my PhD thesis committee members and providing me with fresh-eyed suggestions on my research.

During my candidature, I was continually impressed by my talented and hardworking colleagues in the Melbourne NLP group and NICTA VRL lab. Their valuable suggestions and support were essential to my research. To name them: Jey Han Lau, Marco Lui, Meladel Mistica, Li Wang, Pingping Tan, Ned Letcher, Bahar Salehi, Florian Hanke, Clint Burford, Oliver Adams, Richard Fothergill, Jim Breen, Andrew MacKinlay, Karl Grieser, Willy Yap, Sumukh Ghodke, Spandana Gella, Rebecca Dridan, Yvette Graham, Karin Verspoor, David Martínez, Antonio Jimeno Yepes, and Lawrence Cavedon. Particular mentions go to Meladel and Florian. Reading groups, jogging and cafe explorations in Melbourne are my best memories beyond my busy studies.

I have also been lucky to have had many funding bodies sponsoring my PhD research. The University of Melbourne offered me a great environment and generous funding to pursue my degree. NICTA generously covered most of my conference travel and training costs.

Finally, I would like to thank my family. My wife took good care of me during her stay in Melbourne. My parents brought me up with their copious love. They sacrificed a lot to support me both emotionally and physically. Words cannot express how grateful I am to them, but their love will always sustain me unfailingly.

Chapter 1

Introduction

Social media sites like Twitter and Facebook have become increasingly popular in recent years. They contain huge volumes of user-generated content in the form of text, social connection data, photos and videos. This thesis concerns the preprocessing of social media text data to make it more accessible for NLP tools and downstream applications. In particular, we focus on the effectiveness and efficiency of approaches for Twitter data processing.

1.1 Background and Motivation

Social media data generally refers to content that is created and shared in online communities by users (Kaplan and Haenlein 2010). For instance, short messages in Twitter,[1] social connections in Facebook,[2] photos in Flickr,[3] and videos in YouTube.[4] These various types of social media shape the way people communicate and have become valuable data sources for many user-centric applications such as those shown in Figure 1.1. For instance, many political and social protests have been organised via social media channels. Companies and governments identify public sentiment towards a product or a policy. Disaster management teams monitor user reports on events they witness in real life and share in social media. Also advertisements can be placed

[1] www.twitter.com
[2] www.facebook.com
[3] www.flickr.com
[4] www.youtube.com

more effectively to concerned groups of social media users based on anonymous user profiling. To scope the research, this thesis concentrates on Twitter text data — a popular and readily available social media text source.

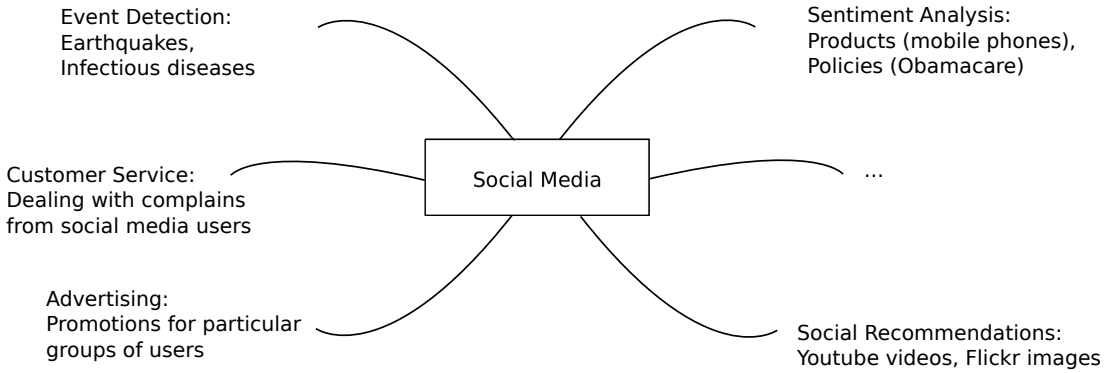


Figure 1.1: Examples of social media-based applications.

Twitter text data is different from conventional text sources such as newswire text. For instance, Twitter messages in Example (1.1) and Example (1.2) are more colloquial and informal than the longer and more formal news text in Example (1.3).

(1.1) Omg phil loves the superbowl omg lol

(1.2) just cuz everyone else is doin it Superbowl in Houston wooooooooo!!!

(1.3) The renowned linebacker, considered by many to be one of the greatest to ever play the game, went out a winner, as the Baltimore Ravens beat the San Francisco 49ers 34-31 in a Super Bowl delayed by a third-quarter power outage.[5]

Specifically, Twitter text is often unedited with an abundance of non-standard words, grammatical and syntactic errors, i.e., Twitter text is generally more *noisy* than newswire text. As a result, it is yet to be effectively harnessed by existing natural language processing (NLP) tools. Existing tools are mainly trained on long, well-formed text data in which word types and sentence structures are largely correct. The noisy Twitter data does not adhere to the standard rules of spelling and grammar,

[5] `http://edition.cnn.com/2013/02/03/us/sports-super-bowl/` (Retrieved 12/2013)

and consequently causes accuracy declines in many existing NLP tools such as part-of-speech (POS) taggers (Gimpel *et al.* 2011) and named entity recognition (NER) taggers (Liu *et al.* 2011b).

The noisy data also hinders the performance of downstream applications. On one hand, many applications require existing NLP tools to process the data, the accuracy decline of these tools on Twitter data will eventually affect the application performance. On the other hand, applications with minimal reliance on NLP also get affected. Taking a Twitter keyword-based event detection system for example, the standard form of "earthquake" has various non-standard variations such as *earthqu*, *eathquake*, and *earthquakeee* — all attested in real Twitter data. These non-standard words will cause inaccurate frequency estimates for keywords, and consequently reduce the utility of a keyword-based system.

Beyond the effectiveness issues, the efficiency requirement is also a pragmatic challenge for Twitter text processing. Existing NLP tools are primarily designed to pursue better accuracy rather than efficiency. The processing speeds of these tools are often not up to the data generation speed (e.g., 500 million Twitter messages per day as reported by Twitter).[6] As a result, the efficiency gap between data generation and consumption speeds also restricts the effectiveness of Twitter data for many real world applications.

1.2 Aim and Scope

Twitter provides massive volumes of user-generated data and the data is notoriously noisy, making it less amenable for existing NLP tools and applications. Motivated by this observation, this thesis targets improving the utility (i.e., effectiveness and efficiency) of social media data in the context of Twitter text processing. Among various potential Twitter text processing treatments, two tasks — text normalisation and geolocation prediction — are identified to bridge the gaps between the existing tools and the Twitter text.

[6] `http://news.cnet.com/8301-1023_3-57541566-93/report-twitter-hits-half-a-billion-tweets-a-day/` (Retrieved 05/2013).

Text normalisation restores non-standard text to its canonical forms so that it can be better recognised and utilised by existing NLP tools. For instance, tokens underlined in Example (1.4) are generally considered as Out-Of-Vocabulary (OOV) words, as they are not seen in the training data of NLP tools in which words are mostly standard entries in a lexicon (i.e., In-Vocabulary (IV) words). If these non-standard words were normalised to their corresponding standard forms as in Example (1.5), then the data becomes more similar to conventional data, and therefore the accuracy of existing NLP tools and applications is expected to improve over Twitter text. The research aim for this task is then to investigate and develop text normalisation methods suitable for Twitter text.

(1.4) btw don't tlk 2 me anymor cuz norml ppl tlk things out they don't sub

(1.5) by the way don't talk to me anymore because normally people talk things out they don't sub

Geolocation prediction disambiguates a public Twitter user's geographical location based on their tweets. The task involves aggregating tweet data from a public Twitter user in order to predict the user's location among a pre-defined set of locations, such as a set of cities. As highlighted with boldface in tweets in Figure 1.2,[7] the prediction often relies on words encoding geospatial information. It includes gazetted terms (e.g., *Melbourne*, *Kyoto*), local politicians (e.g., *Kevin Rudd* is a former Australian Prime Minster), local brands (e.g., *Telstra* is a mobile carrier company name) and local buildings (e.g., *collegesquare Swanston*). Together, these words suggest the potential locations of the Twitter user.

Geolocation information is essential for many social media-based applications. Considering this event detection scenario: Because Twitter is a mixed data stream and contains messages generated across the world, common events (e.g., fire and car accidents) are likely to co-occur and are not distinguished without knowing the geolocation of the event, although rare events like earthquakes and typhoons are less

[7]We highlight the hashtags in the example, but exclude them in our system for generality, as they are primarily applicable to Twitter data.

Figure 1.2: A demonstration of geolocation prediction input and output for a public Twitter user. The red dot icon denotes the predicted city centre.

likely to be affected. Similarly, sentiment analysis performed at the state- or city-level allows more fine-grained user polarity estimation than that conducted at the country level (Schulz *et al.* 2013). For instance, political campaigns and product promotions can be selective based on the user's location, and thus might be more effective. The geolocation information also enables location-based data partitioning, which could benefit many location-specific applications when only a geographically-relevant proportion of the data is desired. For instance, the Twitter data generated from users outside the studied regions are discarded in local event detection. The system then only needs to deal with data from the concerned region, and avoids unnecessary computational cost for irrelevant data. Overall, geolocation information

has the potential to improve both the effectiveness and efficiency of many NLP tools and applications on Twitter text. The aim of this research is thus to develop a geolocation prediction approach, to identify and examine influential factors on the accuracy of geolocation prediction, and to provide guidelines on the design of practical geolocation prediction systems based on research outcomes.

Although this thesis mainly focuses on Twitter text, the methods developed for these tasks are also applicable to other social media data including Facebook updates, and YouTube comments. Furthermore, Twitter specific entities are excluded in the experiments to make the methods more general.

1.3 Contributions

The concrete contributions of this thesis can be largely grouped by text normalisation and geolocation prediction, which consequently improve the effectiveness and efficiency of NLP on Twitter data.

For text normalisation:

- We scope the research by setting up a token-based lexical normalisation task for English Twitter messages, and conduct a pilot study of non-standard words in the data. Based on the observations of non-standard words, we propose a correction generation-and-selection strategy to detect and normalise non-standard words per tokens. We also examine the accuracy of representative benchmarks for approaching the normalisation task, and further discuss the strengths and weaknesses of each method.

- Based on the analysis of the token-based normalisation approach, we propose a pure type-based normalisation approach that directly maps non-standard words to their standard forms using a combined normalisation lexicon. We use both distributional semantics and string similarity to automatically construct a normalisation lexicon. In particular, we experiment with various configurations for these similarities to improve the quality of the lexicon. The derived lexicon is highly complementary to existing normalisation lexicons such as Internet

slang, and by combining these lexicons together, the best published results at the time are achieved on a public normalisation dataset. This pure lexicon-based normalisation approach has reasonable precision and recall. It is also very fast, suitable for preprocessing high volume of noisy Twitter text data. Furthermore, this simple and lightweight lexicon can be easily integrated into a Twitter processing pipeline.

- We evaluate the impact of text normalisation on a downstream Twitter POS tagging task. The results reveal the effectiveness of text normalisation, although the accuracy boost is not comparable to an in-domain Twitter POS tagger.

- Having developed a lexicon-based normalisation approach for English, we then discuss the potential generalisation to other languages. Through experiments on Spanish Twitter normalisation, we demonstrate the generality of the proposed lexicon-based approach, and find a similar complementarity of the automatically-mined lexicon to existing lexicons.

- We also summarise recent text normalisation results that (primarily) use our off-the-shelf combined lexicon in NLP and Twitter-based applications. Progress on text normalisation methods is also discussed and compared, which sheds light on the design of future text normalisation methods.

For geolocation prediction:

- We formulate geolocation prediction as a multi-class classification problem. The prediction then matches a user's tweet data to the most similar tweet data from a location in a pre-defined set of locations (e.g., cities). We provide a unified geolocation prediction framework, under which a range of influential variables for improving the prediction accuracy are addressed and compared, e.g., classification models and feature sets.

- We propose the concept of *location indicative words*, and find that using only these words rather than all tokens in location modelling and inference helps to improve the prediction accuracy. A range of feature selection methods are

applied and compared to identify and select location indicative words. The effectiveness of feature selection has been demonstrated over a number of experiments with different classification models, datasets and location partitions.

- In addition to investigating the location indicative words, we further explore and discuss other influential factors. The extensions fall into two categories: data and methods. On the data part, the comparison between regional and global prediction tasks has implications for the intrinsic difficulty of the task. Furthermore, our exploration of tweeting languages and extra amount of data suggests they both substantially affect the prediction accuracy. As for classification methods, a series of comparisons have been performed, e.g., simple discriminative versus simple generative models, simple versus advanced models, and base classifier versus ensemble methods. The results suggest simple methods are superior, and the influence of model choice is minor, compared to the influence of data in the geolocation prediction task.

- Beyond explorations on tweet text, user metadata is also examined and compared with the tweet text-based method. Experiment results suggest user metadata fields carry varying amounts of geospatial information. In particular, the user-declared location in tweet metadata is a gold mine for geospatial information, and classification using this source achieves the best single text source performance, namely, 40.5% in accuracy. By combining these text sources in ensemble learning, the accuracy of text-based geolocation prediction is substantially improved, achieving the state-of-the-art text-based geolocation prediction accuracy, i.e., 49.1%. This accuracy boost demonstrates the geospatial complementarity of different text sources.

- Twitter is an evolving platform; popular topics may change and new contents may emerge along with time. To better estimate the generalisation of the proposed approach, we evaluate our trained model on time-heterogeneous data to examine the temporal influence on the model. Furthermore, we calibrate predictions based on confidence variables so that only users whose predicted locations

are more reliable are reported. This is because Twitter users are not equally geolocation predictable, and in doing so we can trade off prediction accuracy and recall. In addition, to address the concern of privacy, we break our proposed approach down into different dimensions of contributing factors. The results reveal the geolocatability of users based on their tweeting behaviour, and provide advice for privacy-concerned users in terms of not being geolocated. These empirical results target practical issues relating to geolocation prediction and have implications for the future design of geolocation prediction systems.

A large body of this thesis focuses on the exploration and the development of text processing methods for the two tasks. Due to the availability of pre-existing datasets and downstream systems, only a number of extrinsic evaluations are performed and summarised in the thesis. Nonetheless, the explored tasks target fundamental issues in making effective and efficient use of social media data. With more readily available datasets in this domain, we believe the developed methods and results will play a bigger role in the research community.

1.4 Thesis Outline

The rest of the thesis contains four chapters. In Chapter 2, we discuss literature on social media and two identified text processing tasks in Twitter. After that, text normalisation and geolocation prediction are investigated in Chapter 3 and Chapter 4, respectively. Each chapter describes the task scope, methodology, evaluation and results for the respective tasks. Finally, we conclude the thesis and outline future work in Chapter 5.

Chapter 2

In this chapter, we discuss and compare the characteristics of various types of social media data. Specifically, we focus on introducing Twitter data and justify the reasons why it is chosen for study in this thesis. We then review the literature on two tasks for Twitter text processing: text normalisation and geolocation prediction. In text normalisation, we first demonstrate and analyse

the accuracy declines of various NLP methods on Twitter data. Then the notion of text normalisation and task formulations are addressed. After that we categorise the normalisation approaches into several different paradigms with increasing complexity: spell checking, sequential labelling, machine translation, and system combination. We summarise conventional normalisation approaches and propose several key features that are essential to Twitter text normalisation. In geolocation prediction, we discuss its motivation and challenges in Twitter. Additionally, we distinguish various location representations by type and granularity. The literature of geolocation prediction is partitioned into three categories based on the primary information used in the method, i.e., social network information, text data, or both. We compare different sets of methods by discussing their advantages and limitations, and we also present details for each method in the partitioned group. Finally, we raise some key questions related to improving geolocation prediction accuracy.

Chapter 3

This chapter includes proposed strategies for identifying and normalising non-standard words to their canonical forms for Twitter text. To make the task tractable, we target out-of-vocabulary (OOV) words in English Twitter text relative to an off-the-shelf dictionary, and classify OOVs based on their context fitness to distinguish real OOV words (e.g., *Obama*) and non-standard words (e.g., *tmrw*). For the subset of non-standard words, a small set of correction candidates are generated by morphophonemic clues. Both word similarity and context are then exploited to select the most probable correction candidate for the word. Based on initial experiment analyses, we further notice most longer non-standard words (i.e., word length ≥ 4) have unambiguous corrections (e.g., *4eva* "forever"). We then build a normalisation lexicon by combining distributional and string similarities, and explore important parameters to improve the quality and coverage of the lexicon. The effectiveness of text normalisation is directly evaluated against human annotated datasets, as well as a downstream NLP task: part-of-speech tagging in Twitter. Having discussed lexicon-based

normalisation for English data, we also generalise the method to Spanish Twitter text, because Spanish is similar to English in terms of text processing, e.g., they both use the same basic alphabet and words are largely space separated. The effectiveness of the proposed approach is demonstrated by the strong complementarity of the derived normalisation lexicon to existing lexicons. Finally, we summarise recent progress on text normalisation, and provide suggestions for the future research of this task.

Chapter 4

In this chapter, we investigate and improve the task of text-based geolocation prediction of Twitter users, i.e., predicting a Twitter user's geographical location based on their tweets and profile metadata. We present an integrated geolocation prediction framework and investigate influential factors which impact on prediction accuracy. In particular, we apply and compare an extensive range of feature selection methods to identify location indicative words and use them in location modelling and inference. The effectiveness of location indicative words in geolocation prediction is evaluated under different configurations of datasets, classification models and location partitions. In addition to exploration of the feature set, we extend the investigation in a number of directions, including the incorporation of non-geotagged data, tweeting language influence, user metadata, temporal influence on the model, geolocation prediction confidence and user geolocatability. A range of hypotheses relative to these directions are tested, including: Does adding extra data improve the geolocation prediction accuracy? Does the tweeting language influence the prediction accuracy? What is the contribution of user-declared metadata in geolocation prediction? Will the prediction model remain effective along with time? Can we calibrate the predictions based on system confidence? In addition, how can users protect their privacy in the context of geolocation prediction?

Chapter 5

This chapter concludes the research outcomes of our proposed methods for text normalisation and geolocation prediction. We summarise our work from exist-

ing normalisation methods and a token-based approach to a type-based method, and a discussion of recent progress on text normalisation. For geolocation prediction, we answer the proposed questions relating to influential factors that impact on the geolocation prediction accuracy. Additionally, we discuss the impact of the proposed Twitter text processing tasks relative to our research theme — improving the utility of social media with natural language processing. Beyond our findings, we also identify a number of limitations with respect to our employed approaches and experiment settings, and discuss a range of directions that can be pursued in the future.

Chapter 2

Literature Review

This chapter reviews the literature on social media text processing. First, social media data is surveyed in Section 2.1. In particular, Twitter data is discussed in detail in Section 2.1.1. After that, two text processing tasks on effectively and efficiently utilising social media data are examined: text normalisation in Section 2.2 and geolocation prediction in Section 2.3. Finally, a brief summary of the chapter is presented.

2.1 Social Media

Social media is an imprecise concept. As noted by Kaplan and Haenlein (2010), it is by and large a set of Internet-based applications that enable user-generated content (UGC) to be created and shared in online communities.

Social media (as shown in Table 2.1) varies in the user engagement and content type.[1] For instance, many encyclopedia articles in Wikipedia are voluntarily created and maintained by domain experts; shared photos in Flickr are taken by both professional photographers and amateurs; social relationships in Facebook and short messages in Twitter are widely adopted by organisations and general users.

Compared with conventional media like television and newspapers, social media

[1] An aperiodically updated list can be found in `http://en.wikipedia.org/wiki/List_of_social_networking_websites` (Retrieved 05/2013)

Social media type	Representative sites and URLs
Wikis	Wikipedia (`http://www.wikipedia.org/`)
Photo Sharing	Flickr (`http://www.flickr.com/`)
Video Sharing	YouTube (`http://www.youtube.com/`)
Social Networks	Facebook (`https://www.facebook.com/`)
Blogs	WordPress (`http://wordpress.com/`)
Microblogs	Twitter (`https://twitter.com/`)

Table 2.1: A partial list of social media sites.

has some compelling features:

1. Social media data is massive in terms of users and the volume of user contributed data. For instance, active users in Facebook had exceeded 1 billion by September 2012. Facebook would have the world's 3rd largest population, if it were a country.[2] The data generated by these active users is enormous.

2. Social media data is created and shared by members of the general public. Unlike conventional media (e.g., newspapers written by journalists), social media also represents voices from general users. For instance, they share photos and videos that they have taken in their personal lives, or they set up connections and create posts to make their voices heard, e.g., the revolutionary protests in the Arab Spring.[3] Despite the gender and age skewness among users (Sage 2013) — there being more younger users than senior users of social media — the right to create and spread ideas is generally not thwarted. Furthermore, because the data comes from the general public, it covers a broad spectrum of topics that are interesting to users, and is beyond the reach of conventional media, e.g., users share events experienced in their daily lives, be it a concert, party or sports game. Users also convey their likes and dislikes towards a tar-

[2] `http://www.independent.co.uk/life-style/gadgets-and-tech/news/revealed-the-third-largest-country-in-the-world--facebook-hits-one-billion-users-8197597.html` (Retrieved 05/2013)

[3] `http://en.wikipedia.org/wiki/Arab_Spring` (Retrieved 05/2013)

get. For instance, they express opinions on political leaders (e.g., Obama vs. Romney), or electronic products (e.g., iPhone).

In summary, social media, as a platform for information creation and dissemination among a vast number of users, is shaping the way of communication.

The huge amount of user-generated data in social media drives many new applications. It has attracted increasing attention from industry, government and the research community. For instance, companies use social media to promote products and advertising, e.g., videos in YouTube to improve product awareness (Hoffmann and Fodor 2010). Consultants analyse user sentiment on particular targets, e.g., products and celebrities (Jiang *et al.* 2011). Emergency sectors monitor social media for reported crises to improve situational awareness (Vieweg *et al.* 2010; Goolsby 2010; Yin *et al.* 2012). Governments may track public responses on new policies (e.g., public sentiment over US foreign policy in presidential election debates (Hu *et al.* 2012)). Researchers exploit social media to make predictions on stock trends (Bollen *et al.* 2011), influenza outbreaks (Ritterman *et al.* 2009; Paul and Dredze 2011), detect disastrous events such as earthquakes (Sakaki *et al.* 2010) and first mentions of breaking news stories (Petrović *et al.* 2010; Petrović *et al.* 2012), and retrieve tweets containing user opinions for a topic (Luo *et al.* 2012).

Social media also poses challenges in data utilisation. The reverse side of the massive amount of data is information overload (Hiltz and Turoff 1985), when a single user's information digestion speed is much slower than the information production speed. Appropriate filtering and searching in a sea of information may partially alleviate this problem. Nevertheless, thousands of Twitter messages are generated per second for popular events, e.g., the Super Bowl (a football game in US),[4] and this rate is still far beyond a user's reading speed.

The data in social media ranges from longer formal sentences to short informal text snippets. Sentences in Wikipedia articles, as shown in Example (2.1), are usually carefully edited. In contrast, Flickr tags and comments tend to be short and colloquial, such as *Spectacular*, *Great shot!*. In Twitter and Facebook, formal sentences,

[4] `https://blog.twitter.com/2014/sb48-253-million-tweets-18-billion-impressions-according-to-nttr` (Retrieved 02/2014)

informal text snippets and uninformative strings co-exist. For instance, while Example (2.2) is a well-formed tweet similar to conventional newswire data, the tweet in Example (2.3) contains many non-standard words that require a readers' non-trivial effort to comprehend, e.g., *C U 2Nyt* represents "see you tonight". Compared with other tweet examples, Example (2.4) is almost meaningless without context information.

(2.1) The Super Bowl is the annual championship game of the National Football League (NFL), the highest level of professional American football in the United States, culminating a season that begins in the late summer of the previous calendar year.[5]

(2.2) The announcement of Pope Francis Wednesday caused a Super Bowl-like stir on the Web.

(2.3) SUPER BOWL SUNDAY!!! Enjoy yourselves!!! Sunday morning GOODIES R sent out! C U 2Nyt!

(2.4) GOOD woooooo LOL

In addition to the huge volume and varying quality of social media data, data availability is also a crucial issue (Eisenstein 2013b). Due to privacy issues and hosting organisation policies, only a few data sources are readily available for constant and large scale analysis, e.g., Wikipedia and Twitter. This thesis primarily focuses on Twitter data for the following reasons: Although many social media corpora are available, the data scale and accessibility is not comparable to Twitter data. Another reason for our interest in Twitter data is its many unique characteristics, which are discussed below in Section 2.1.1.

2.1.1 Twitter Data

Twitter is a popular microblogging service which allows millions of users to create and share short text messages (also known as tweets). It can be conveniently accessed

[5] `http://en.wikipedia.org/wiki/Super_bowl` (Retrieved 05/2013)

via multiple devices such as desktop computers and mobile phones. The volume of Twitter text is growing at a staggering speed. For instance, the number of tweets posted per day in 2009, 2010, and the first half of 2011 were 2, 65 and 200 million, respectively.[6] This number had exceeded 500 million by late 2012.[7]

While social media data is created by users, the data is most typically hosted by companies. One compelling advantage that Twitter has over other social media is its open policy to research. Twitter offers two major application programming interfaces (APIs) for accessing the public data shared by users:

- Streaming API samples public tweets and pushes them to an authenticated user end point. The connection is long-lived, which is convenient for users to harvest real-time tweets. This API samples between 1% and 10% of all tweets generated by public users, depending on the access level granted by Twitter.

- RESTful API supports fine-grained access and interactions with Twitter. For instance, one can request to only obtain the profile information from a group of specified users. It also allows an application to send data back to Twitter, e.g., to send a direct message to a friend. Nonetheless, a common use of the RESTful API is ad hoc retrieval searching for tweets matching a particular query such as obtaining tweets with the keyword *apple* from all users. The RESTful API is rate-limited, but one can do post-analysis over tweets by archiving data from the Streaming API.

Apart from open access to the data, tweets also have an extra real-time property that many other social media sources do not. For instance, Wikipedia usually mirrors data monthly, while posted tweets can be accessed within seconds, e.g., via the Streaming API. This property is essential for real-time applications such as event detection.

In the rest of this section, we first take a text-centric view of Twitter, and then discuss the social networking aspects of this service.

[6] `https://blog.twitter.com/2011/200-million-tweets-day` (Retrieved 05/2013)

[7] `http://news.cnet.com/8301-1023_3-57541566-93/report-twitter-hits-half-a-billion-tweets-a-day/` (Retrieved 05/2013)

Text in Twitter

A (downloaded) tweet is a JSON format object,[8] including both tweet text and metadata. The text is strictly limited to 140 characters, and content beyond that length is truncated. While the text is written by users across the world in many languages, English is the most frequently used language, accounting for approximately half of the whole data (Hong *et al.* 2011; Baldwin *et al.* 2013). The quality of Twitter text varies. Newswire-like texts indeed exist in tweets as shown in Example (2.5). However, due to the free-form nature of much tweet text, many non-standard words are also observed in tweets. These non-standard words include typos (e.g., *earthquick* "earthquake"), abbreviations (e.g., *ppl* "people") or phonetic approximations (e.g., *2nyt* "tonight") as shown in Examples (2.6)–(2.7). On top of these non-standard words, tweets also contain non-standard sentence structures as shown in Example (2.8). Additionally, spam (e.g., Example (2.9)) also accounts for a non-trivial proportion of tweets, e.g., approximately 8% of URLs in tweets are spam (Grier *et al.* 2010).[9]

(2.5) With high youth unemployment in parts of Australia, are young migrants still finding work?

(2.6) I wish we can sleep 2nyt then wake up tmrw mornn and see dat ds whole madness is over.

(2.7) Tell ppl u luv them cuz 2morrow is truly not promised.

(2.8) So clever-jay!!wont stop using Glasses!In hz vidz and fotos.

(2.9) http://su.pr/1UNYgk earn more money today read this

Tweet text also contains Twitter-specific entities such as user mentions, hashtags, and URLs. A user mention entity (as shown in Examples (2.10)–(2.11)) starts with an "@" symbol followed by a character sequence and ends with a space separator.[10]

[8] `http://www.json.org/`

[9] The spam statistics are changing over time, however, little update can be found in recent papers (up until 05/2014). Recent literature has focused more on anti-spam approaches rather than reporting new statistics.

[10] Due to privacy concerns, all users names are anonymised in our examples.

It can be a linguistic constituent in a tweet as shown in Example (2.10), or it can be just a user account that is involved in the tweet discussion, be it a real user account or a virtual organisation. Similarly, a hashtag starts with a "#". It can be part of the sentence as in Example (2.12), but it is often used to identify a topic or a conversation as in Example (2.13) (Tsur and Rappoport 2012). In order to save space, URLs in Twitter are often shortened. For instance, `http://edition.cnn.com/2013/06/03/us/boston-bombing-victim/index.html` may be replaced by `http://goo.gl/Z7d5D`.

(2.10) Congrats @USER What an epic match!

(2.11) Li Na is so solid. Deep groundies. Confident movement and shot selection. Serve the trigger. Up 4-0, after love service. @USER

(2.12) Graduating this week? Share your #uomgrad moment and be in the running to win a $500 wardrobe thanks to @USER - http://ow.ly/rGthP

(2.13) Follow @USER today for more coverage of the gun violence petition delivery to Congress. #NotBackingDown #EarlyFF

Beyond the tweet text, Twitter data is also accompanied with rich metadata in the JSON payload. The metadata offers extra information about tweets and users such as the time when a tweet was posted and time zone information. It is worthwhile to mention that some fields are not always available. For instance, the location field in a Twitter user profile may be empty if a user doesn't specify it.

Twitter also enables user interactions in tweets, e.g., to reply a tweet or email it to friends. One Twitter specific operation is called retweet, represented by "RT" in a tweet. RT is an explicit symbol that usually signifies that a user has re-posted a tweet. It could be to indicate support for the information content of the tweet, or for forwarding the information to other users.

Social Relationships in Twitter

Social relationships in Twitter are more complex and diverse, compared with other social networks (e.g., Facebook) in which mutual friendship is the primary

network information. *Following* (e.g., fans following a celebrity) is a uni-directional social relationship in Twitter. This asymmetric relationship is often not equivalent to friendship. One example is fans follow a celebrity to find out about them, and while some celebrities have local fan bases, more popular celebrities tend to have fan bases which are spread across the globe. In contrast, *reciprocal following* relationships (i.e., bi-directional mutual followings between two users) are much more indicative of a friendship.

In addition to these explicit and well-defined social relationships, there are also implicit social relationships that can be recovered from the data in Twitter.[11] User mentions, tweet favourites, tweet replies and retweets are implicit asymmetric social interactions. These interactions can be further grouped to construct symmetric relationships. For instance, user mentions in tweet conversations indicate a symmetric social friendship between *USER1* and *USER2* as shown from Example (2.14) to Example (2.16).

(2.14) USER1: New MacBook Pro: Red battery indicator. Still 1:22 to go. #yay

(2.15) USER2: @USER1, @USER3 I feel jealous of your guys. I'd like to see your new Mac.

(2.16) USER1: @USER2 You will soon! (Not much to see, though. Not even pixels) /cc @USER3

Twitter social relationships differ in strength and availability. While *reciprocal followings* are strong indicators of online social ties, the tie strength of an asymmetric retweet is much weaker, e.g., consider a fan retweeting a tweet from a popular celebrity. Explicit relationships (e.g., reciprocal following) can always be retrieved, provided the user account is not protected. In contrast, implicit relationship availability is subject to a user's tweeting habits, e.g., whether they prefer to include user mentions in tweets.

[11]Different from Twitter's documentation which refers to *following* as friendship, only mutual followings are considered to indicate friendship in this thesis.

2.1.2 Summary

In Section 2.1 , we reviewed social media in terms of its features, challenges and applications. In particular, we examined Twitter data in detail in Section 2.1.1. Different from conventional corpora (e.g., newswire text), Twitter is a noisy data source and the data volumes are huge. These features may hinder the data utilisation by existing NLP tools and applications. To enable effective and efficient use of the data, the next two sections discuss two Twitter text processing tasks, which can be summarised as a "divide and clean" treatment.

On "cleaning the data", text normalisation in Section 2.2 focuses on converting various non-standard words in social media to their canonical forms, e.g., from *4eva* to "forever". By doing so, the normalised data is expected to be more tractable for existing NLP tools.

In the sense of "dividing the data", geolocation prediction in Section 2.3 enables data partitioning by location. It makes location-based applications feasible and also avoids dealing with large amounts of irrelevant data, as a means of improving data use efficiency.

2.2 Text Normalisation

In this section, we first introduce non-standard words and the factors that play a role in their formation in Section 2.2.1. Furthermore, we examine the impact of non-standard words on a number of NLP tasks, showing their negative influence on existing tools. After that, we review the text normalisation task in Section 2.2.2 and discuss normalisation methods using different metaphors in Section 2.2.3. Because new normalisation methods are constantly emerging, most recent literature is separately discussed in Section 2.2.4, and the discussion of this recent literature is placed in Section 3.6 in comparison to our work. Finally, we summarise the section and outline the desired proprieties in normalising social media data in Section 2.2.5.

2.2.1 Non-standard Words

Non-standard words exist in most texts, from proofread newswire to unedited social media messages. For instance, *SSN* is commonly used to denote "Social Security Number" in various sources in the United States. Complex spellings are also approximated in web search queries, e.g., "Schwarzenegger" is sometimes spelt *Scwartegger* as noted by Cucerzan and Brill (2004). Mobile phone SMS messages often contain non-standard words like *cu* "see you" and *ttyl* "talk to you later". With the increasing popularity of social media, Twitter also contains a plethora of non-standard words, including typos (e.g., *simliar* "similar"), phonetic approximations (e.g., *4ever* "forever"), words with repetitions (e.g., *soooo* "so") and informal abbreviations (e.g., *srsly* "seriously"). These non-standard words may be intentionally typed (e.g., using *SSN* for brevity) or unintentionally generated (e.g., "similar" is mistakenly typed as *simliar*).

Formation of Non-standard Words

It is difficult to enumerate all possible factors underlying the formation of non-standard words. Nonetheless, a number of reasons have been identified (Jones 2010; Eisenstein 2013b), including:

- **Spelling**: Regardless of English literacy, when people cannot remember the correct spelling, they usually mimic word spellings based on morphophonemic clues (Choudhury *et al.* 2007; Eisenstein 2013a), e.g., *earthquick* "earthquake" and *overwelm* "overwhelm".

- **Convenience**: Some users tend to create and use shortened non-standard words in place for longer words and phrases when typing on mobile devices, e.g., *tmrw* "tomorrow" and *ttyl* "talk to you later" are popular abbreviations in Twitter.

- **Text length limits**: Similar to convenience reasons, some social media services have hard word limits, e.g., the maximum number of characters per message is

140 per tweets. As a result, users have to shorten standard words to accommodate more information in one message.[12] As shown in Example (2.17), *Kor* and *Jpn* are abbreviated forms of Korean and Japanese.

(2.17) all the issues that arose XD and we helped them translate so much for foreign fans lol between my friend & I we had Kor Eng Chi Jpn covered

- **Input devices**: The resultant non-standard words are also impacted by tweeting devices (Gouws *et al.* 2011b). Many Twitter users send messages via mobile devices. The limitations of the input method can cause users to save keystrokes when typing, e.g., *u* "you". Additionally, some mobile devices can be configured to automatically correct typos, e.g., typing *Adress* may be corrected to "Address". Such features make a tweet message less susceptible to spelling errors, however, it may also introduce new errors due to overcorrection.

- **Emphasis**: In contrast to word shortenings, character repetitions often embody the polarity and the degree of sentiment from a tweet author (Brody and Diakopoulos 2011). For instance, *looooove* "love" in Example (2.18) conveys strong positive sentiment. Similarly, sarcasm can be expressed in repeated characters such as *looooove* in Example (2.19).

(2.18) First and for most happy birthday to @USER looooove you darling

(2.19) It's a good thing Waianae doesn't have alternate routes to get in and out. Since I just looooove sitting in traffic! #sarcasm

- **Community**: Users from a particular community tend to adopt certain non-standard words as a form of social marking, e.g., *yolo* "you only live once" is often used by younger people.[13] Users outside this community are often unable to determine the deabbreviated meaning. Non-standard words used by

[12]Eisenstein (2013b) has demonstrated the text limit factor is not the primary cause of non-standard words in Twitter, however, there indeed exists tweets that are affected by the text length limit such as Example (2.17).

[13]`http://www.washingtonpost.com/blogs/arts-post/post/yolo-the-newest-abbreviation-youll-love-to-hate/2012/04/06/gIQA3QE2zS_blog.html` (Retrieved 05/2013)

Twitter users also vary across different ethnic backgrounds (Eisenstein 2013a) and geographical regions (O'Connor *et al.* 2011). For instance, *jus* "just" is popularly used in African American English in the United States.

These surveys are primarily based on Twitter text. Because Twitter covers a broad spectrum of non-standard words (Eisenstein 2013b), we expect many of the findings to also apply to other text types (Baldwin *et al.* 2013), e.g., blog and forum data. Having explained potential reasons for the formation of non-standard words, we now discuss the impact of non-standard words on natural language processing.

The Impact of Non-standard Words

To make use of social media data for various applications, reliable text processing tools are required, e.g., to learn a statistical model from text or to examine its effectiveness on the text. Substantial accuracy declines of existing tools have been noted over a number of NLP tasks when they are applied to Twitter, e.g., part-of-speech tagging (Gimpel *et al.* 2011; Owoputi *et al.* 2013), syntactic dependency parsing (Foster *et al.* 2011), named entity recognition (Liu *et al.* 2011b; Ritter *et al.* 2011), and machine translation (Wang and Ng 2013). We discuss these accuracy degradations by analysing real tweet examples.

Part-of-speech (POS) taggers are usually trained and evaluated on single sentences sourced from edited text such as the Wall Street Journal section of the Penn Treebank (Marcus *et al.* 1993). Most widely used POS taggers (Brill 1992; Schmid 1994; Ratnaparkhi 1996; Toutanova *et al.* 2003) are based on lexicalised features, e.g., modelling a conditional probability $P(t|w)$ or a joint probability $P(t, w)$ for a POS tag t and a word w. Because non-standard words in tweets are often not recognised, accurate conditional probabilities cannot be estimated (other than through smoothing). Consequently, non-standard words are likely to be incorrectly tagged. As shown in Example (2.20), a raw tweet with non-standard words (labelled with (a)) and its manually revised version (labelled with (b)) are independently tagged by the Stanford POS tagger (Toutanova *et al.* 2003).[14] *2nite* is incorrectly labelled as an adjective

[14]Stanford English POS tagger version 3.1.5

rather than a noun due to the non-standard phonetic approximation of "tonight". This negative impact can be further aggravated when there are many non-standard words in a tweet as shown in Example (2.21). *ppl*, *u* and *luv* are all labelled as nouns in tweet (a). If those words were mapped to their standard forms as in tweet (b), the POS tagger then outputs the correct labels.

(2.20) (a) Will/MD I/PRP see/VB you/PRP 2nite/JJ ?/.

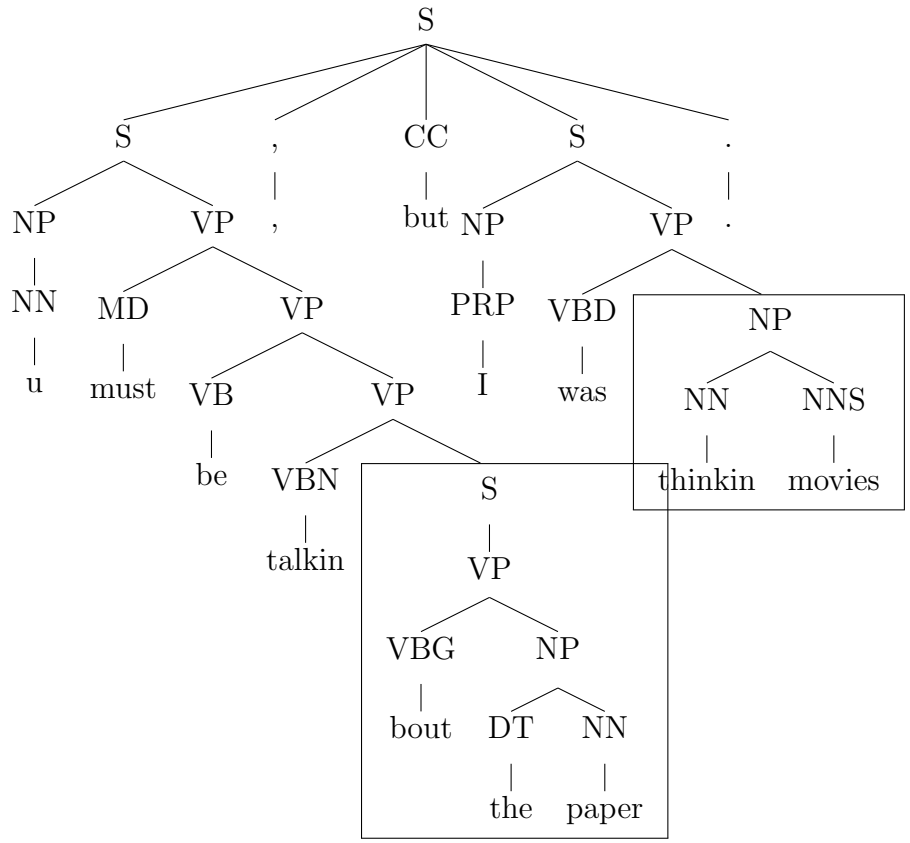
(b) Will/MD I/PRP see/VB you/PRP tonight/NN ?/.

(2.21) (a) Tell/VB ppl/NN u/NN luv/NN them/PRP cuz/VBP 2morrow/NNS is/VBZ truly/RB not/RB promised/VBN ./.

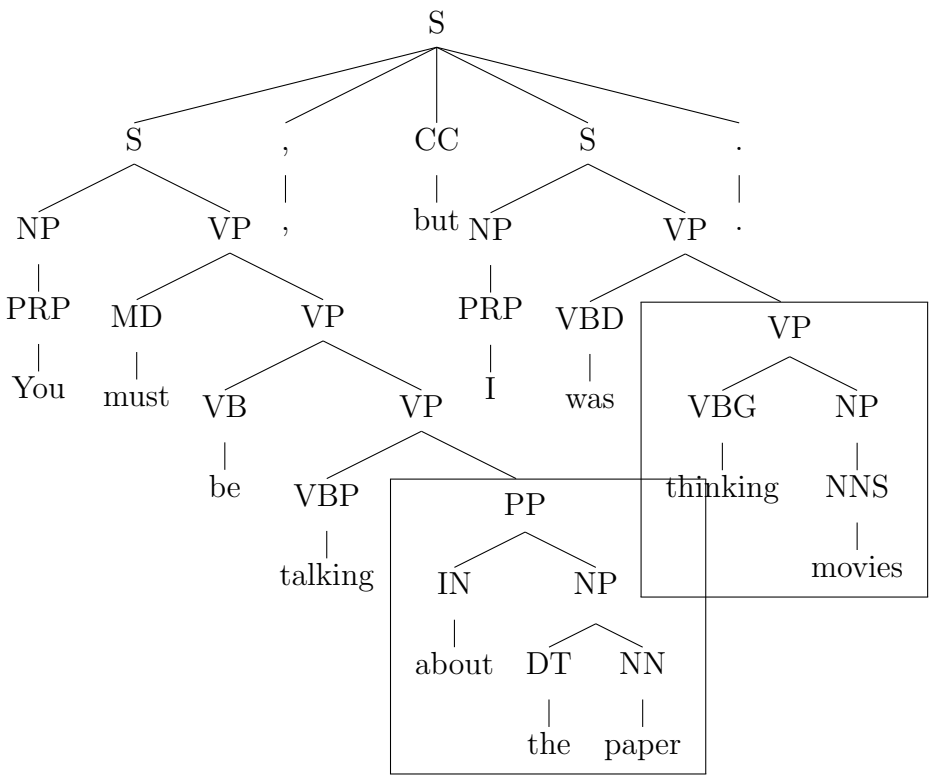
(b) Tell/VB people/NNS you/PRP love/VBP them/PRP because/IN tomorrow/NN is/VBZ truly/RB not/RB promised/VBN ./.

Syntactic parsing accuracy is also affected by non-standard words as shown in Example (2.22). The Stanford English parser (Klein and Manning 2003; De Marneffe *et al.* 2006) analyses *bout the paper* and *thinkin movies* as a clause and noun phrase, respectively, rather than a prepositional phrase and verb phrase. If the non-standard words were correctly normalised as in Example (2.23), the parser then outputs the correct result.

(2.22)


S
S
,
CC
S
.
NP
VP
,
but
NP
VP
.
NN
MD
VP
PRP
VBD
NP
u
must
VB
VP
I
was
NN
NNS
be
VBN
S
thinkin
movies
talkin
VP
VBG
NP
bout
DT
NN
the
paper

(2.23)


S
S
,
CC
S
.
NP
VP
,
but
NP
VP
.
PRP
MD
VP
PRP
VBD
VP
You
must
VB
VP
I
was
VBG
NP
be
VBP
PP
thinking
NNS
talking
IN
NP
movies
about
DT
NN
the
paper

Recent parsing experiments on forum data (Foster 2010) and a variety of unedited text sources (Baldwin *et al.* 2013) suggest that non-sentential constituency, erroneous tokenisation and non-standard words are the main error sources in parsing. In Example (2.22), the input text is correctly pre-tokenised by the Stanford parser's built-in tokeniser, and the sentence is also structurally well-formed, i.e., there are no missing constituents, although word orthographies may be non-standard. Hence, these parsing errors seem to be due to non-standard words. It is not surprising that many lexicalised Probabilistic Context Free Grammar (PCFG) parsers yield poor results in such contexts, as words are encoded in grammar rules. However, the Stanford parser utilises an unlexicalised PCFG to generate syntactic grammar rules without using detailed lexical information which means the parsing step is less sensitive to concrete non-standard word types. Nonetheless, the Stanford parser relies on the output of a POS tagger, and as we analysed above, non-standard words cause troubles for POS tagging. As evidenced by studies on German (Petrov and Klein 2008) and Arabic (Marton *et al.* 2010), the interaction between POS tagging and parsing suggests POS tagging errors can be amplified in parsing. This indicates the parsing errors in Example (2.22) are mainly caused by unreliable POS tagging outputs. In summary, non-standard words cause errors in POS tagging, and the errors further propagate to syntactic parsing.

In machine translation, the challenge is similar to POS tagging. Non-standard words do not match entries in the phrase table, and are thus not translatable. As shown in Example (2.24), *earthquick* is left intact in an English to Chinese translation using Google Translate.[15] If the tweet were correctly normalised in Example (2.25), the Chinese translation would be complete and correct.

(2.24) Help us , Avril !! This is the biggest earthquick in JAPAN
帮助 我们 ， 艾薇儿 ！ 这 日本 最大 earthquick

(2.25) Help us , Avril ! This is the biggest earthquake in JAPAN
帮助 我们 ， 艾薇儿 ！ 这 是 日本 最大的 地震

[15] `http://translate.google.com/` (Retrieved 05/2013)

The negative impact of non-standard words can also be amplified in machine translation. In Example (2.26), a noisy English phrase *have lunch 2gether* is broken down into word-level Chinese translations as 吃 午饭 *2gether* 的 (which consists of a collection of tokens: *eat*, *lunch*, *2gether*, and a redundant auxiliary word 的). In contrast, when the phrase is normalised to "have lunch together" in Example (2.27), the correct translation 共进午餐 is generated.

(2.26) we finally got 2 have lunch 2gether
我们 终于 拿到了 2 吃 午饭 2gether 的

(2.27) we finally got to have lunch together
我们 终于 共进午餐

Capitalisation has been found to be essential for named entity recognition (NER) in both edited text (Arnav Khare 2006) and Twitter data (Ritter *et al.* 2011). Because of the informal register, tweets often have a higher ratio of incorrect capitalisations than edited text sources, and this causes more troubles for tweet NER (Liu *et al.* 2011b; Ritter *et al.* 2011). We compared NER results using an off-the-shelf tool.[16] The identified named entities for (a) original and (b) case-normalised tweets are shown in Examples (2.28)–(2.29). As we see in Example (2.28), when words are in uppercase, the identification of named entities becomes difficult and consequently results in false positives, e.g., the NER tool incorrectly suggests *CANT BR3ATHE* as a person name, but omits the true person name *OBAMA* at the beginning. Conversely, when tweets are in lowercase, many correct named entities are omitted. For instance, *obama* and *machel obama* are considered to be normal content words in Example (2.29).

(2.28) (a) OBAMA'S SPEECH WRITER. I [PER CANT BR3ATHE]

(b) [PER Obama] 's speech writer. I cant br3athe

(2.29) (a) @USER hi president obama tell machel obama i said hi and i just want to say go obama family

[16] `http://cogcomp.cs.illinois.edu/demo/ner/results.php` (Retrieved 05/2013)

(b) @USER hi president [PER Obama] tell [PER Machel Obama] I said hi and i just want to say go [PER Obama] family

It is worthwhile noting that non-standard words are an important but not the only source of noise that affects the accuracy of NLP tools in Twitter. Twitter entities also complicate NLP tasks. As demonstrated in Examples (2.10)–(2.11) on Page 19, hashtags may be linguistic constituents or topic markers. Furthermore, the same hashtag may also play different roles depending on context. For instance, the hashtag in Example (2.30) is a syntactic subject. In contrast, the same hashtag in Example (2.31) is a topical tag and is not part of the syntax of the sentence. Similarly, a tweet with incomplete sentence structure could be problematic. Example (2.32) highlights the difference of a tweet (a) with a missing subject and (b) its manually revised form, when POS tagging using the same Stanford POS tagger.

(2.30) #icwsm is the official hashtag for ICWSM. Conference begins in 12 hours!

(2.31) All welcome! Meet 12:40 in Kresge Foyer. #icwsm

(2.32) (a) Goin/NN home/NN this/DT weekend/NN

(b) I/PRP am/VBP going/VBG home/RB this/DT weekend/NN

As demonstrated by the aforementioned examples, the effectiveness of NLP tools is greatly hindered by non-standard words and other noise in social media text, because most NLP tools are designed to be trained and evaluated on data drawn from *independent and identical distributions* (i.e., from the same domain). When the test data is different from the training data (e.g., in terms of words, register and sentence structures), the accuracy of existing tools often declines. This is largely known as a *domain adaptation* problem (Daumé and Marcu 2006; Blitzer *et al.* 2006). To reduce this effect and improve the utility of social media data, NLP research has taken two approaches: (1) adapt NLP tools to social media text, for instance, to annotate data and define new features with respect to social media text for POS taggers (Gimpel *et al.* 2011; Owoputi *et al.* 2013), dependency parsers (Foster *et al.* 2011) or named entity recognisers (Liu *et al.* 2011b; Ritter *et al.* 2011); and (2) normalise text to

their standard orthography in English (Aw *et al.* 2006; Liu *et al.* 2011a; Xue *et al.* 2011), Spanish (Alejandro Mosquera and Moreda 2012) and Chinese (Xia *et al.* 2006; Wang and Ng 2013), and then apply conventional NLP tools. Because this thesis focuses on processing Twitter data, we review the second approach (i.e., text normalisation) in detail in the next section. Additionally, we compare these two approaches on a POS tagging task in Section 3.4.

2.2.2 Normalisation Task and Scope

The concept of text normalisation was first proposed by Sproat *et al.* (2001) in the context of a preprocessing step for text-to-speech conversion. For instance, *bdrm* and *apt* are not readable without appropriate treatment in Example (2.33), because word-level phonetic transcriptions for these non-standard words are not available in text to speech systems, which are often trained on formal, edited datasets. However, if the two non-standard words were normalised to "bedroom" and "apartment", respectively, it would be much easier to read them appropriately in speech synthesis (Schwarm and Ostendorf 2002).

(2.33) In 1988, a four bdrm apt only costs $1M.

In general, text normalisation transforms non-standard words into their contextually appropriate canonical forms, making the data more amenable for downstream processing (Sproat *et al.* 2001).[17] This is a vague definition, but it is also very flexible, allowing for application dependent normalisation. For instance, a keyword-based event detection system might require normalisation of misspellings (e.g., *shakin* "shaking"), informal abbreviations (e.g., *dis mgt ctrs* "disaster management centres") and phonetic approximations (e.g., *earthquick* "earthquake") for accurate keyword counting. Similarly, *2014* can be pronounced differently, depending on whether it represents a year or is a cardinal number. While NER is sensitive to capitalisations, syntactic parsing is straightforwardly affected by incorrectly split characters (e.g., *l o v e* "love")

[17]Originally, text normalisation was defined to include sentence tokenisation, and the detection, categorisation, and restoration of non-standard words. This thesis focuses on detection and restoration of non-standard words.

and concatenated words (e.g., *cu* "see you"). In addition, restoring missing punctuation and sentence constituents (e.g., subjects) may also help to improve tweet readability for humans. Nonetheless, most work on text normalisation primarily focuses on non-standard words consisting of alphanumeric characters (Cook and Stevenson 2009; Beaufort *et al.* 2010). These words can be categorised into four types, as shown in Figure 2.1.

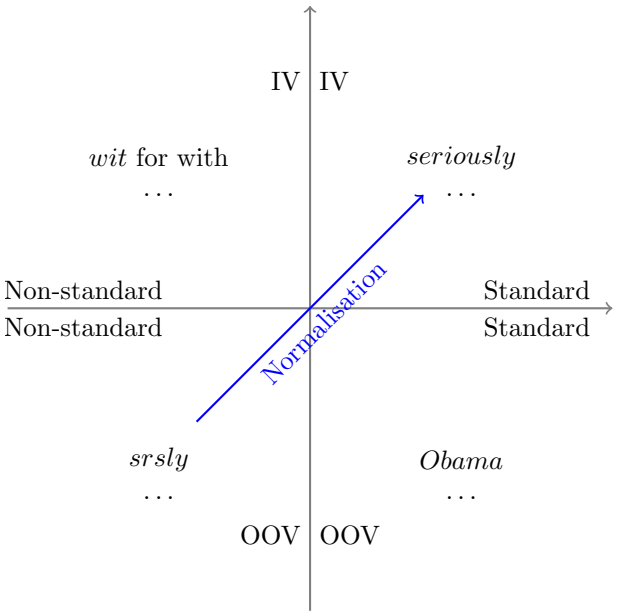


Figure 2.1: Word categorisations in text normalisation.

Non-standard words include both Out-Of-Vocabulary (OOV) non-standard words and In-Vocabulary (IV) non-standard words. Both types of non-standard words differ from their standard forms and can take some effort and context for humans to comprehend. On the one hand, many non-standard words are OOV words, although they often correspond to standard words in tweets, e.g., *srsly* represents "seriously" in Example (2.34).

(2.34) $50 for shopping is srsly not enough, and i'm not even kidding

On the other hand, some non-standard forms of IV words happen to coincide with other IV words, however, they usually do not fit the context, e.g., *wit* in Example (2.35).

(2.35) I will come wit you

The detection of non-standard words is challenging for both types. IV non-standard words are computational expensive to identify, because every token in the text must be examined. Due to this pragmatic issue, OOV non-standard words receive more attention in text normalisation and IV non-standard words are often ignored. OOV words are relatively easier to identify, by checking whether they are present in a given lexicon. However, not all OOV words are non-standard words. There are many named entities which are not included in lexicons, but are nevertheless standard forms, e.g., *Obama*. The classification of OOV words as standard or non-standard is not trivial. As such, many normalisation systems either assume non-standard words have already been identified (Liu *et al.* 2011a), or largely treat all OOVs as non-standard words (Sproat *et al.* 2001).[18]

When discussing mapping non-standard words to their standard forms in normalisation, another unresolved issue is what defines a standard form? While *talkin* is non-standard and its standard form is "talking", whether *IBM* is a standard word or it should be normalised into "International Business Machines" is arguable. To make the normalisation task more tractable, a standard form is often solicited from an IV lexicon. This lexicon can be based on a commonly-used off-the-shelf dictionary. Alternatively, a corpus-derived lexicon from the target domain can also serve the purpose, e.g., all types with token frequency $\geq$ some threshold in a particular corpus. In both approaches, whether *IBM* is an IV word or should be normalised is then naturally settled.

Text normalisation has been tailored to various granularities relative to the normalisation scope. One straightforward normalisation approach is to manipulate characters within non-standard words to revert them to their canonical forms (i.e., context-insensitive lexical normalisation), e.g., dropping repetitive characters in *hooooot* "hot" and adding a missing *g* in *takin* "taking". In some cases, this word-centric character manipulation is insufficient to capture ambiguous non-standard words in the data. For instance, *hw* represents "how" in Example (2.36) and "homework" in Example (2.37).

[18]Sproat *et al.* (2001) also included common abbreviations and rule patterns to improve the detection performance for non-standard words, however, the detection is largely based on a lexicon.

(2.36) Hi, hw are you?

(2.37) Let me finish my hw though

To deal with this uncertainty, context information beyond the target non-standard word is considered in normalisation (i.e., context-sensitive lexical normalisation). "homework" makes more sense than "how" as the standard form of *hw* in Example (2.37), because "finish my homework" is more likely than "finish my how" in terms of trigram frequency. The scope of these normalisations focuses on non-standard words, and each non-standard word is independently normalised as in conventional spell checkers. As a result, these methods are referred to as spell checking-based approaches, as discussed in Section 2.2.3.

Nevertheless, normalisations for multiple non-standard words may be mutually influenced when context words are non-standard words as well. For instance, *yr* can be interpreted as "you're" or "year" in Example (2.38). If the second non-standard word *srs* were normalised to "serious", then the chance of *yr* being "you are" should become higher.

(2.38) Oh wow yr srs

To capture the mutual influence of normalisations for adjacent non-standard words, joint normalisation for the whole sentence is preferable. In this setting, the canonical forms are not independently selected for each non-standard word, but the decisions are made by optimising the likelihood of normalisations over the whole sentence. This more flexible and powerful approach is often interpreted as a sequential labelling task.

In most cases, a non-standard word is normalised to a single canonical form. However, non-standard words can be the result of splitting and concatenating standard words. As a result, non-standard words may be grouped together for many-to-many normalisations, e.g., *I l o v e i t* "I love it" and *cu* "see you". Additionally, text normalisation may also involve the insertion of missing words and the deletion of redundant words, such as deleting *I* in Example (2.39). These powerful normalisations are often modelled as a monolingual machine translation task that translates

noisy text with non-standard words to standard English text in the target domain, as addressed in Section 2.2.3.

(2.39) I I swear all my friends have boyfriends and I'm like ..... Oh

2.2.3 Methodologies

Text normalisation aims to find the most appropriate canonicalised text t for a noisy text s. It has similarities with a range of existing tasks, e.g., spell checking, query log correction, and SMS and forum data cleansing. In most cases, the task can be formulated in a probabilistic framework as finding $\arg\max_t P(t|s)$, which can be further interpreted via a unified noisy channel model (Kemighan *et al.* 1990; Li *et al.* 2006; Aw *et al.* 2006), as shown in Equation (2.40).

$$\arg\max_t P(t|s) = \arg\max_t \frac{P(s,t)}{P(s)} = \arg\max_t \frac{P(s|t)P(t)}{P(s)} \tag{2.40}$$

Because $P(s)$ is common for all t, $P(t|s)$ is proportional to the likelihood $P(s|t)$ and the prior $P(t)$. $P(t)$ is often estimated using a language model built from target-domain corpora. $P(s|t)$ characterises the formation of non-standard words from their canonical forms, which is often hard to capture, as discussed in Section 2.2.1. As a result, Equation (2.40) is approximated with various assumptions and variations.

Spell checking aims to find the most similar IV word to replace each misspelling in the sentence. Given a noisy text s of length n, i.e., $s = (s_1, s_2, \ldots, s_n)$, the likelihood $P(s|t)$ is then expanded as in Equation (2.41). This formulation assumes the independence of words in s, and that s_i is only influenced by t_i.

$$\begin{aligned} P(s|t) &= P(s_1|t)P(s_2|t)\ldots P(s_n|t) \\ &= P(s_1|t_1)P(s_2|t_2)\ldots P(s_n|t_n) \\ &= \prod_{i=1}^{n} P(s_i|t_i) \end{aligned} \tag{2.41}$$

Note that the expansion can be formulated in different ways such as making a particular normalisation conditioned on the nearby context words, e.g., $P(s_i|t_{i-1}, t_i, t_{i+1})$.

Instead of normalising non-standard words independently, $P(s,t)$ can be factored as a hidden Markov model (HMM) (Rabiner 1989) to enable collective normalisation for the whole sentence. Equation (2.42) demonstrates a second-order HMM.[19] The emission probabilities are similar to Equation (2.41). In contrast, the transition probabilities additionally model the influence of adjacent normalisations in which the two proceeding normalisations, t_{i-2} and t_{i-1}, affect the current normalisation t_i.

$$\begin{aligned} P(s,t) &= P(s_1, s_2, \ldots, s_n, t_1, t_2, \ldots, t_n) \\ &= \underbrace{\prod_{i=1}^{n+1} P(t_i|t_{i-2}, t_{i-1})}_{\text{transition}} \underbrace{\prod_{i=1}^{n} P(s_i|t_i)}_{\text{emission}} \end{aligned} \tag{2.42}$$

The variants of the noisy channel model are so far largely based on one-to-one word mappings.[20] A more flexible machine translation approach allows many-to-many normalisations from a noisy sentence s of length m to a normalised sentence t of length n (Brown *et al.* 1993).[21] The flexibility is essentially due to the introduction of an extra alignment factor a as in Equation (2.43). a maps each word in s to words in t. $P(a_i|i,m,n)$ represents the probability of the ith alignment in s for a given noisy text of length m and a potential normalisation of length n.

$$\begin{aligned} P(s|t) &\propto P(s,a|t) \\ &= P(s_1, s_2, \ldots, s_m, a_1, a_2, \ldots, a_m|t_1, t_2, \ldots, t_n) \\ &= \prod_{i=1}^{m} P(s_i|t_{a_i}) P(a_i|i,m,n) \end{aligned} \tag{2.43}$$

These variations provide a general idea of how normalisation is tackled in different contexts. In the following sections, we discuss seminal references relative to each methodology.

[19] t_{-1}, t_0 and t_{n+1} are special starting and ending symbols.

[20] They can be configured to support one-to-many normalisations, e.g., by enabling missing word recovery (Wang and Ng 2013)

[21] We demonstrate the formulation with IBM model 2.

Spell Checking-based Approach

The history of computer-based spell checking dates back to the 1960s, when the focus was on correcting typographical errors (Blair 1960; Damerau 1964) and optical character recognition (OCR) errors in converting images to text (Takahashi *et al.* 1990). These errors (as one type of non-standard word) are mainly caused by one letter insertion, deletion, or substitution, or the transposition of two letters (Damerau 1964). Fortunately, most errors are recoverable from one or a few character edits. Peterson (1980) discussed pragmatic issues related to the concept, design and implementation of a spell checker. Strictly speaking, a spell checker only identifies whether an examined word is a misspelling or not. However, it is often also equipped with a correction function that substitutes the misspelling with the correct word. Similar to OOV non-standard words and IV non-standard words, misspellings can be categorised as non-word errors and real-word errors (Kukich 1992; Golding and Roth 1999; Hirst and Budanitsky 2005), respectively. Most work focuses on non-word errors due to the same efficiency reason discussed in Section 2.2.2.

Corresponding to Equation (2.41), $P(s_i|t_i)$ is often calculated by the similarity between s_i and t_i (Ahmad and Kondrak 2005), and this similarity can be defined in various ways. For instance, edit distance is one of the most widely used measurements to calculate the minimum number of character edits to transform an IV word t_i to a misspelling s_i. Beyond character variations, phonetic similarity is also considered, e.g., using Soundex[22] or Double Metaphone (Philips 2000). These algorithms generate abbreviated phonetic "signatures" based on s_i and t_i. Words with the same signature are phonetically more similar than words with different signatures. For instance, *allwaz* and *always* have the same signature in Double Metaphone, although the surface forms of the two words are different.

Nonetheless, the accuracy of such methods is far from perfect. For instance, Damerau (1964) suggested that around 80% of errors can be corrected by these methods in formal texts, however this number is based on unintentional typos, and excludes various types of non-standard words found in social media. Furthermore, these sim-

[22] `http://www.archives.gov/research/census/soundex.html`

ple methods can be further improved to attain better accuracy. For instance, the probabilities of letter transitions are different, e.g., *ch* is more likely to be substituted by *sh* than *ie* (Zobel and Dart 1996).

To bridge the gap, Brill and Moore (2000) improved the basic edit distance by estimating fine-grained edit probabilities. Paired correct words and misspellings are decomposed into possible character segments. The optimal segments are aligned in the Expectation-Maximization (EM) algorithm. By doing so, the likelihood (i.e., $P(s_i|t_i)$ in Equation (2.41)) is approximated by the product of segment probabilities. For instance, *belief* is aligned with *bilif* in Example (2.44), then $P(bilif|belief) = P(b|b) \times P(i|e) \times P(l|l) \times P(i|ie) \times P(f|f)$.

(2.44)

	belief	b	e	l	ie	f
		↓	↓	↓	↓	↓
	bilif	b	i	l	i	f

The segment probabilities can be further conditioned on the segment positions in the word, e.g., at the beginning, middle or end of the word. The intuition is that the chance of e being mistakenly substituted by i is higher when e is in the middle of the typo (e.g., *relivant* "relevant") than when it occurs at the beginning (e.g., *igg* "egg"), based on Brill and Moore's (2000) empirical observations. To increase the alignment quality, the segment generation allows up to two edits for the aligned segments, e.g., s in *birsday* can be aligned to "th" in "birthday".

Toutanova and Moore (2002) raised the inadequacy of Brill and Moore's (2000) grapheme-based model in correcting some phonetic errors, e.g., *saing* is corrected as "sang" instead of "saying". To remedy the deficiencies, they proposed to incorporate a phoneme-based model which performs grapheme-to-phoneme conversions for both non-standard words and paired normalisations. The phoneme-based model operates on the same training data, but adds extra phonetic constraints on the edits. The final normalisation is selected based on a log-linear combination of grapheme-based probability and phoneme-based probability.

Ahmad and Kondrak (2005) corrected query log spelling errors in an unsupervised manner, i.e., without relying on paired misspellings and corrections. First, they identified a set of IV query words relative to a query token as the correction candidates

using edit distance-based approximate string matching. This leads to a substantial reduction of correction candidates relative to the query misspellings. Then, they formulated the similarity between a query and its correction candidate as a sequence of character edit probabilities. The character edit probabilities are estimated over a corpus of queries using the EM algorithm. The IV word with the highest edit probabilities is then selected as the correction for the given query token.

Additionally, distributional similarity (Lin 1998) has also been exploited to correct query misspellings (Li *et al.* 2006). The IV word sharing the most similar context with a misspelling (relative to a background corpus) is selected as the correction. Likewise, a non-standard word in tweets is often used in similar contexts to its correct normalisation. As shown in Examples (2.45)–(2.48), *tmrw* and its normalisation "tomorrow" are largely interchangeable. Different from the edit distance-based method that compares the surface form similarity, a distributional similarity-based model estimates $P(s_i|t_i)$ by leveraging context information. The final $P(s_i|t_i)$ is estimated by a linear combination of the two models.

(2.45) See you tmrw then (:

(2.46) Hey guys, see you tomorrow. Good night!

(2.47) I'm going to be so tired tmrw morning..

(2.48) I am going to be sooo tired tomorrow morning...

Many non-standard words greatly deviate from their standard forms, which is often beyond the scope of a spell checker. Word lengthening is relatively easy to detect, e.g., *goooood* "good". In contrast, severe word shortenings (e.g., *b4* and *srsly*) are often not captured by spell checkers designed for more conventional text.[23] Therefore, much attention has been paid to capturing these new creative forms in SMS and social media. Cook and Stevenson (2009) proposed multiple error generation models, each of which captures a particular way in which SMS non-standard words

[23]E.g., correct normalisations for these two words are not obtainable via the Jazzy spell checker (`http://sourceforge.net/projects/jazzy/`).

are formed, such as phonetic spelling (e.g., *epik* "epic") and clipping (e.g., *walkin* "walking"). Xue *et al.* (2011) integrated four similar formation models, but adapted the error formation to Twitter data. For instance, they incorporated frequently used acronyms (e.g., *cu* "see you") as one of their models.

Sequential Labelling-based Approach

Text normalisation can be formulated as a sequential labelling task to capture the mutual influence of word-level normalisations. The input is a tokenised raw text s. For each token in the text, similar IV words are generated as normalisation candidates. The normalised text t is the candidate sequence that maximises $P(t|s)$. The selection of candidates is often achieved by the Viterbi algorithm based on a language model built from the target domain. This process equates to the selection of tags (i.e., IV words) for each token in sequential labelling.

Cucerzan and Brill (2004) corrected query log misspellings using a weighted edit distance and a language model. The weighted edit distance assigns unequal probabilities to different character edits, e.g., dropping the final g in a word is more probable than deleting g at the beginning of a word. The weights are estimated using query log statistics. They further assumed IV tokens are more frequent than the corresponding misspellings. Therefore, the correction is performed by substituting rare tokens with similar but more frequent tokens in a query, e.g., from *Amzon* to *Amazon* in *Amzon kindle*. The selection of corrections for the whole query follows the standard Viterbi algorithm relative to a bigram language model. Similarly, Contractor *et al.* (2010) proposed to generate normalisation candidates by observing patterns of similarity between non-standard words and candidates. In particular, they used the ratio of longest common character subsequence over the consonant edit distance to obtain potential normalisation candidates for non-standard words. The candidates are also selected using the Viterbi algorithm to form the most likely normalised sentence.

Choudhury *et al.* (2007) exploited a hidden Markov model (HMM) to normalise SMS text. They captured two types of errors — cognitive errors and typos — in one model. Cognitive errors are primarily caused by errors in the spelling process

in a user's mind. Under this model, a word is spelt based on meaningful character segments, and then these segments are typed either in their morphological forms or their phonetic approximations. Typos occur when a user makes unintentional errors typing the spelling in their mind, such as missing a *t* when typing *committee*. As a result, the whole process (i.e., from forming the sentence to typing it on the screen) consists of spelling in the mind and then typing segments. These two parts are then modelled as HMM transitions and emissions, respectively.

Zhu *et al.* (2007) introduced a unified Conditional Random Field (CRF) model to normalise informal text at different granularities. The informal text ranges from excessive line breaks between paragraphs to inappropriate casing of a given word. They normalised these informal phenomena in a sequential labelling framework, and used labels like ALC (i.e., all lower case for characters in a given word) and DEL (i.e., delete a line break) to denote the edits to the informal text. The normalisation primarily focuses on stylistic normalisation such as restoration of capitalisation and redundant punctuation elimination. The study excludes the normalisation of non-standard words. This is partly due to the data sparsity issue in supervised learning methods, i.e., it is expensive to obtain a large amount of annotated data for correcting various typos and informal abbreviations. Furthermore, the percentage of non-standard words in the study (i.e., newsgroups) is relatively small. Nonetheless, the methodology can be shifted to non-standard word normalisation if sufficient training data becomes available in the future.

Beaufort *et al.* (2010) tackled French SMS normalisation using finite state machines. Tokens in SMS messages are first divided into fine-grained character segments based on the character-level alignments in the training data, which consists of noisy SMS texts and their normalised counterparts. Conditioned on whether the token is an IV word or an OOV word, these segments are further transformed by re-writing rules derived from the alignments. After that, the re-written segments are combined and mapped to possible word sequences. An SMS trigram model is applied to select the optimal word sequence as the normalised SMS. Nonetheless, this method requires large-scale labelled training data, which is often not available for Twitter.

Liu *et al.* (2011a) designed customised queries to obtain noisy word-level training data from Google search results, e.g., *tmorro* "tomorrow". Both non-standard words and potential corrections are aligned at the character-level using longest common subsequence information. These alignments are then used to train a character-based CRF model to capture the likelihood of a word being transformed into a noisy token. Together with a word unigram model, the normalisation of a noisy token is estimated by the product of a normalisation's prior and its likelihood to be converted to the noisy token as formulated in Equation (2.40) on Page 34. A similar CRF model is also used by Pennell and Liu (2011b), in which they focus on normalising deletion-based non-standard words, e.g., *srsly* "seriously".

Machine Translation-based Approach

A more ambitious view of text normalisation is to consider the task as monolingual machine translation from noisy text with non-standard words to standard English sentences in the target domain.

Aw *et al.* (2006) applied a phrase-based machine translation method (Koehn *et al.* 2003) to normalise non-standard words for SMS text. They leveraged annotated data which consists of noisy and clean paired sentences to train the translation model. The phrases are aligned using the EM algorithm. Similarly, Kaufmann and Kalita (2010) also adopted supervised phrase-based machine translation for tweets. Their approach further incorporated simple preprocessing steps, including removal of repeated letters in non-standard words before feeding the text into a machine translation system. Both experiments demonstrate the utility of supervised machine translation to improve the translation BLEU score (Papineni *et al.* 2002).

Instead of applying phrase-based machine translation in normalisation based on learning translation phrases from parallel data, Pennell and Liu (2011a) proposed to use character-based machine translation. The character-based system is advantageous because it avoids data sparsity issues of word- or phrase-based systems. Fine-grained character-level variation is captured and applied to non-standard words that are not found in the training data. For instance, if *fite* "fight" is in the training data and

nite "night" is absent, word-based systems are unlikely to correct *nite* to "night" as a character-based system would, due to lack of fine-grained knowledge of character-level translations (i.e., *te* to "ght").

Normalisation System Combinations

To further improve the normalisation accuracy, an intuitive approach is to combine different systems together. Many approaches integrate multiple existing normalisation systems in one way or another.

Kobus *et al.* (2008) analysed the strengths and weaknesses of existing text normalisation approaches. For instance, while spell checking makes use of lexical similarities, it is largely performed on the basis of individual words, and often ignores context-sensitive misspellings. Likewise, machine translation is able to handle many-to-many normalisations, but it is hampered by the data sparsity issue, that is, creative non-standard forms are unlikely to be captured by supervised machine translation without abundant training data, which is typically not available. Having compared various approaches, Kobus *et al.* (2008) combined a customised machine translation system with a speech recognition approach (i.e., sequential labelling) for SMS normalisation. A noisy SMS is firstly translated into relatively clean text using machine translation, then the OOV words in the text are further segmented and mapped to phoneme sequences. These phoneme sequences are re-assembled into word sequences using a phoneme-to-word lexicon. Finally, the best word sequences are selected to form the final output relative to a trigram model.

Gao *et al.* (2010) utilised a universal normalisation candidate ranker to correct misspellings in query logs. For each query and correction candidate pair (s, t), various surface form similarity and frequency features are converted into real values and stored in a feature vector $\mathbf{f}$. The feature vector is then transformed into a feature function score y in a linear model $y = \mathbf{w} \cdot \mathbf{f}$, in which $\mathbf{w}$ is the weight vector and is optimised using paired misspellings and corrections. y indicates how likely t is the correction of s, i.e., $P(s|t)$. They further added phrasal machine translation probabilities in the feature vector like other similarity features. In addition, a range of web-scale

language models are trained and applied to estimate $P(t)$.

Gouws *et al.* (2011a) applied a cascaded framework for Twitter text normalisation. A small automatically-derived lexicon is used to capture common and frequent lexical variants such as *ppl* "people". When constructing the lexicon, they first use distributional similarity to derive contextually-similar type pairs. These pairs are then filtered relative to an IV lexicon, and only pairs that consist of an OOV and IV word are preserved. The preserved pairs are further re-ranked by the ratio of overlapping subsequences, and the top-50 highest pairs are selected for the lexicon. For the remaining non-standard words, they adopted a sequential labelling method to generate normalisation candidates and decode the normalisation sequence.

Liu *et al.* (2012) improved the recall of a character-based CRF model (Liu *et al.* 2011a) using extra features and system combination. They cleaned the noisy training pairs using cosine similarity of their context words and added rich morphophonemic features (e.g., phonemes, syllables, word boundaries) in the CRF modelling. Furthermore, they incorporated a human *visual priming* factor in the model. The *visual priming* favours the candidate with the highest frequency and the longest common subsequences with the noisy token among all normalisation candidates. Additionally, candidates are required to share the same starting character as the noisy token. The normalisation is performed by combining an enhanced CRF tagger, *visual priming* and a spell checker. As different normalisation modules produce different candidates, the approach improves the recall of normalisation at the cost of a relatively small candidate number. Having sourced various normalisation candidates, the final normalisation is also achieved by Viterbi decoding relative to a n-gram model.

Li and Liu (2012) proposed to use phonetically similar character segments instead of individual characters for normalisation. This is because alignment based on character segments is more plausible than alignment based on individual characters, e.g., *ph* is aligned to *f* in Example (2.50), rather than *p h* to *f null* in Example (2.49). Furthermore, decoding a word into character segments results in shorter decoding length than using individual characters. Having obtained the aligned character segments from the training data, they applied both sequential labelling and machine translation to restore segments in test data to word sequences. They found the best

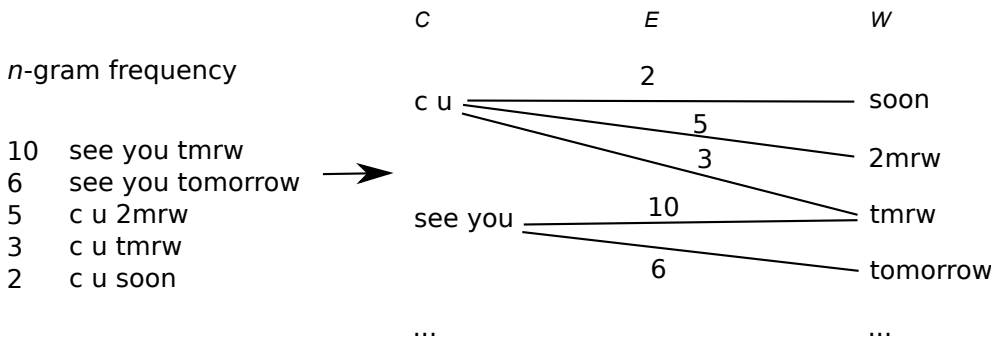


Figure 2.2: From n-grams to a bipartite graph $G = (W, C, E)$.

accuracy is achieved when combining existing systems together, i.e., a spell checker, character-level and character segment-level machine translation models, and a character segment-level sequential labelling method.

(2.49)

photo	p	h	o	t	o
	↓	↓	↓	↓	↓
foto	f	null	o	t	o

(2.50)

photo	ph	o	t	o
	↓	↓	↓	↓
foto	f	o	t	o

2.2.4 Recent Normalisation Approaches

In addition to methods discussed in Section 2.2.3, many important text normalisation approaches have emerged in the last two years. We review the technical details in this section, and leave the discussion of these approaches to Section 3.6.

Hassan and Menezes (2013) incorporated random walks from lexical variants to standard words in text normalisation. As shown in Figure 2.2, n-grams are extracted from a massive Twitter corpus, and subsequently decomposed into the target words (W) and the context (C), forming a bipartite graph $G = (W, C, E)$. E are connections between W and C, and are weighted by the n-gram frequency in the tweet data. W contains both standard words and lexical variants.

Once G is built, random walks from lexical variants to standard words are repeated many times with respect to a pre-defined maximum hop limit. As a result, each lexical variant has a distribution of standard words which is shaped by the co-occurrence of

the context and the target, i.e., "tomorrow" is a more likely random walk destination than "soon" for *tmrw*, supposing *c u soon*, *c u tmrw*, *see you tmrw* and *see you tomorrow* occur 2, 3, 6 and 10 times, respectively. Instead of using the standard word with the highest probability as the normalisation, the top-n standard words are used as normalisation candidates for each lexical variant. To form the final normalised sentence, these candidates are jointly selected using the Viterbi algorithm relative to a language model trained on clean text from the target domain.

Zhang *et al.* (2013) normalised tweets for the purposes of improving syntactic parsing. Each token in a raw tweet is replaced by normalisation candidates derived from a range of sources, e.g., spell checking and Internet slang lexicons. For instance, *ii*, *wll* and *cu* are replaced by {is, I, ...}, {well, will, ...} and {see, see you, ...} in *ii wll cu soon*. IV words are left untouched, e.g., *soon* in the example. These normalisation candidates are then jointly synthesized to the most probable sentence using dynamic programming relative to a language model built from the target domain.

The proposed approach benefits from flexible candidate generation and global optimisation. On the one hand, it integrates various existing candidate generation methods (e.g., based on edit distance and spell checker suggestions), rather than solely relying on the context derived candidates. Furthermore, the proposed normalisations can be more than one word, e.g., *cu* is expanded to "see you". This is particularly important, because a missing constituent can be detrimental to parsing (Foster 2010). On the other hand, joint decoding of the normalisation sequence takes contextual information into account, which is superior to the spell checker-based methods that primarily focus on individual lexical variants. For instance, both "well" and "will" can be normalisations of *wll*. The preceding normalisation "I" (for *ii*) in a bigram language model suggests "will" is more likely the correct normalisation, because "I will" is more probable than "I well".

Yang and Eisenstein (2013) also formulated tweet normalisation as a sequential labelling task, and exploited a unified log-linear model to capture string and context similarities between any lexical variants ($w_{oov} \in V_{oov}$) and standard words ($w_{iv} \in V_{iv}$). While this reduces the human effort in carefully engineering string and context similarity features, the challenge is computational tractability and efficiency. For

example, not every (w_{oov}, w_{iv}) pair can be observed, even though the tweet data is massive in size. Furthermore, each token in a tweet can be replaced by a standard word from a dictionary V_{iv} which is typically larger than 10^4 words in size. As a result, the Viterbi decoding is impractical even for short tweet text, as it takes $O(n|V_{iv}|^2)$ time to calculate the optimal normalisation sequence for a tweet of n tokens. To tackle this issue, they adopted sequential Monte Carlo methods to sample plausible normalisation candidates relative to the target language model, i.e., instead of searching the whole V_{iv} for normalisations, they selected a subset of plausible candidates V'_{iv} from V_{iv}, and then performed the same Viterbi decoding. The size of selected V'_{iv} is used to trade-off normalisation accuracy and efficiency.

Ling *et al.* (2013) leveraged bilingual and monolingual tweets to build a two-layer normalisation system. First, they constructed a training dataset consisting of noisy and clean tweet pairs from parallel bilingual corpora using online translation systems. They assumed the translation of a non-English tweet is the cleaned-up version of the original English string. The assumption is based on the observation that lexical variants in English often do not have counterparts in the paired non-English tweets. As a result, translating the non-English tweet into English may harvest normalisations for lexical variants in the paired original English tweet. In Example (2.51) for instance, (a) and (b) is a parallel English-Chinese tweet pair. The original English tweet contains the lexical variant *tmrw*, which has a standard Chinese translation "明天". By translating it using an online translator from Chinese to English, the correct normalisation form "tomorrow" is then presented in (c). As a result, (a) and (c) form a silver-standard training pair.

(2.51) (a) Getting VERY excited about tmrw morning. Go @USER!!! http://ow.ly/7nnGB

(b) 对明天早晨非常兴奋，去@USER!!! http://ow.ly/7nnGB

(c) Very excited for tomorrow morning, go to @ USER!!! Http://ow.ly/7nnGB

Having obtained the silver standard training data, both phrase- and character-based machine translation systems are trained and applied in normalisation. First,

a character-based system transforms individual lexical variants to canonical counterparts. After that, the phrase-based system selects the optimised normalisation sequence at the sentence level. As a supervised learning task, the coverage of training data is an important issue. To tackle this issue, they incorporated context-similar noisy pairs in the character-based system, e.g., given *tmrw* "tomorrow" is derived from the training data and *tmrw* and *2mrw* have similar context of usage, *2mrw* "tomorrow" is also considered as a normalisation pair, which has a similar effect to the random walk-based approach of Hassan and Menezes (2013).

Similarly, Xu *et al.* (2013) leveraged redundancy in tweets to derive large-scale comparable monolingual data (i.e., noisy and clean pairs) for normalisation. They first clustered tweets with overlapping named entities and temporal expressions. Then they paired noisy and clean tweets in the cluster relative to a language model trained on `NYT`. These paired tweets are further filtered by setting a minimum Jaccard distance (Lee 1999) and a minimum sentence length for tweet pairs. Having obtained the comparable tweet pairs, they incorporated these pairs in a machine translation system for normalisation.

Wang and Ng (2013) adapted a machine translation decoder in normalisation. The decoder operates in two dimensions. When performing normalisation, hypotheses are generated for each token from the beginning to the end of a raw tweet. A proposed hypothesis is a partially normalised tweet rather than a candidate word as in previous methods. This setting is essential for the second dimension operations, in which each proposed hypothesis (i.e., partially normalised tweet) is evaluated using lightweight features such as language model scores. The evaluated hypotheses are further pruned using beam search, and only the highest-scored partially-normalised tweet is kept until all tokens are examined. A hypothesis can be generated in various ways, such as using phonetic approximations from *4* to *for*. The generation process can also flexibly incorporate common punctuation (e.g., comma, periods) and crucial missing *be* verb restoration.

Chrupała (2014) approached Twitter text normalisation using CRF-based edit sequences and string transformations wrapped in simple recurrent networks. Different from previous approaches that adapt generative Bayesian methods, the author

proposed to tackle text normalisation as direct editing non-canonical data into the standard forms. The edits, such as character insertions and deletions, are generated in a linear chain CRF model. The simple recurrent network brings string transformation information from the unlabelled tweet text in a way that is similar to conventional language model, e.g., generate a character based on the activations of the hidden units.

2.2.5 Summary

With the proliferation of computer-mediated communication, non-standard words are becoming increasingly prevalent in short message services (SMS), Internet Relay Chats (IRC), online forums and social media data like Twitter and Facebook. These non-standard words reduce the accuracy of existing NLP tools which are often trained on edited text with few non-standard words. To improve the utility of social media data, text normalisation reduces the lexical variance in the data, and the cleaned data is expected to be more accessible to existing NLP tools and downstream applications.

Text normalisation strives to transform non-standard words to contextually appropriate forms (Sproat *et al.* 2001). The task setting is configurable, and in many cases, it maps OOV non-standard words to their standard IV orthographies as shown in Figure 2.1 on Page 31. The need for normalisation is application dependent and can be performed at different granularities. Instead of examining each approach chronologically, we organised and categorised methodologies into four groups based on their primary ideas:

- Spell checking-based methods focus on single non-standard word substitution, based on morphophonemic similarity and/or context information. These methods can be implemented in an unsupervised manner, and thus are suitable for normalising Twitter text, because it is often hard to obtain training data in Twitter normalisation. However, compared with other approaches, they are less flexible in normalisation. For instance, many-to-one mappings such as *l o v e* "love" are generally not covered.

- Sequential labelling-based methods consider the mutual influence of adjacent normalisations for non-standard words. This type of approach can be considered as a sequential generalisation of spell checking-based methods. Instead of normalising each non-standard word individually, normalisation candidates are generated for each non-standard word. These candidates are then selected based on the overall likelihood of the normalisation sequence. Compared with spell checking-based methods, sequential labelling-based methods improve the flexibility of normalisation and only require an extra language model built from the target domain to select the optimised normalisation sequence.

- Machine translation-based approaches translate noisy text with non-standard words to clean text. They are flexible in enabling many-to-many normalisations and can also handle concatenated and split non-standard words. In addition, they can be configured to include missing word restoration and redundant token elimination. However, the weakness of these methods is they usually require large amounts of appropriate data to train the system. Furthermore, machine translation-based methods involve a series of processing steps such as word alignment. Hence, they are often "heavier" and slower than the previous two types of methods in terms of computation.

- System combination of the above methods further enhances text normalisation. It leverages the advantages of the base methods to achieve better accuracy, although the normalisation efficiency is the slowest among all methods.

Overall, from spell checking-based methods to system combination, text normalisation becomes increasingly powerful and meanwhile more technically challenging. Approaches for text normalisation are evolving, and advanced methods are constantly being proposed.

Text normalisation for Twitter data has many pragmatic challenges. First, the amount of Twitter data is huge, and normalisation must occur in a timely manner, otherwise, the normalised data would be less useful for time-critical applications like real-time event detection. Furthermore, it is crucial to distinguish OOV non-standard

words (e.g., *tmrw*) from standard OOVs (e.g., *Obama*) before performing normalisation in an end-to-end normalisation system. Moreover, many powerful and flexible normalisation approaches are supervised methods such as machine translation-based methods. Given that the types of non-standard words are diverse in social media, they usually require non-trivial amounts of training data to achieve accurate results. However, this data is not readily available in Twitter, and is often expensive to construct. Consequently, semi-supervised or unsupervised methods that leverage large volumes of unlabelled data are more suitable for Twitter normalisation. To address these challenges, we present our explorations on Twitter text normalisation in Chapter 3.

2.3 Geolocation Prediction

In this section, we first present the motivations and challenges in Twitter geolocation prediction in Section 2.3.1. After that, we categorise related work by the primary information utilised in prediction models. Social network-based methods and text-based methods are discussed in Section 2.3.2 and Section 2.3.3, respectively. Hybrid methods integrating various sources of geolocation information are presented in Section 2.3.4. This section primarily highlights seminal references on geolocation prediction. More relevant literature to our work is presented in Chapter 4. Finally, we summarise existing Twitter geolocation prediction literature in Section 2.3.5.

2.3.1 Background

While acknowledging potential privacy concerns of exposing a user's location to the public (Mao *et al.* 2011; Pontes *et al.* 2012), accurate geographical location prediction is a key driver for location-specific services. For instance, search engine providers can offer more suitable results tailored to users' regions (Gravano *et al.* 2003). Similarly, advertisements can be placed according to the target locations. Furthermore, influenza detection (Sadilek *et al.* 2012b; Dredze *et al.* 2013) and natural disaster event detection (Sakaki *et al.* 2010; Yin *et al.* 2012) have been shown to benefit from geolocation awareness. Given geospatial information, emergency response and rescue

operations can be more effectively coordinated. In addition, geolocation prediction enables better interpretation of user sentiment from different regions (Schulz *et al.* 2013). For instance, candidates of political elections are keen to know the public sentiment across different states to campaign more effectively. Geospatial information also helps to identify words and topics in Twitter that are salient for particular regions (Eisenstein *et al.* 2010; Yin *et al.* 2011; Hong *et al.* 2012; Dalvi *et al.* 2012; Ahmed *et al.* 2013) which are potentially valuable to assist lexicographers to compile a regional dialect lexicon. In summary, the awareness of geolocation enables a plethora of location-based applications, and provides a geospatial dimension for data analysis and interpretation.

Given the importance of geographical information, location prediction has been the target of research across different disciplines over the last decade. For example, tagging the geolocation of user queries (Wang *et al.* 2005b; Backstrom *et al.* 2008; Yi *et al.* 2009), blogs (Fink *et al.* 2009), and web pages (Ding *et al.* 2000; Amitay *et al.* 2004; Zong *et al.* 2005; Silva *et al.* 2006; Bennett *et al.* 2011) has been considered in information retrieval. In geographical information science, the primary focus has been on recognising geographical references in text (Quercini *et al.* 2010; Leidner and Lieberman 2011). In case the references are ambiguous, they must be resolved to unique locations (Leidner 2007; Ireson and Ciravegna 2010), e.g., to determine that a mention of *Melbourne* refers to Melbourne, Australia or alternatively Melbourne, US. Within the social media realm, geolocation methods have been applied to Flickr images (Crandall *et al.* 2009; Serdyukov *et al.* 2009; Hauff and Houben 2012; O'Hare and Murdock 2013; Laere *et al.* 2013b), Wikipedia articles (Lieberman and Lin 2009; Adams and Janowicz 2012), Facebook users (Backstrom *et al.* 2010), tweets (Kinsella *et al.* 2011; Priedhorsky *et al.* 2014), and Twitter users (Eisenstein *et al.* 2010; Cheng *et al.* 2010; Kinsella *et al.* 2011; Wing and Baldridge 2011; Roller *et al.* 2012). Compared with tweet-level geolocation prediction, Twitter user geolocation is usually more accurate and reliable, because it leverages all tweets from a user, which offers more information than a single short tweet.

Categorisation of Geolocations

Before we discuss geolocation prediction methods, it is essential to discuss the granularity and the type of geolocations (Wang *et al.* 2005a).

In terms of location granularity, a user's geolocation ranges from precise points (e.g., GPS coordinates) and fine-grained place names (e.g., office room number in a building) to coarse-grained cities, states and countries. Fine-grained locations are rarely used in the literature, because data containing fine-grained locations is often insufficient, except in some densely populated areas (Sadilek *et al.* 2012a). Furthermore, fine-grained user location prediction raises strong privacy concerns, even when the data is publicly accessible. For instance, users may feel unconformable when the house number and street name of their home is publicly revealed. In contrast, it is less disturbing to geolocate users at the city- or country- level when they are willing to share their tweets. By merging data from the same coarse-grained location, more abundant information is available to distinguish between different locations. For these reasons, much existing work investigates city-level geolocation, e.g., to distinguish whether a user's geolocation is in London, UK or New York, US.

Beyond the location granularity, the types of locations can be categorised into *about location*, *tweeting location*, and *primary location*. We illustrate the differences between these location types using tweet examples from real Twitter accounts.

(2.52) Oh you know, just trying to hold up the Effiel tower

(2.53) @USER I live in California. Bay Area

The *about location* refers to places that a tweet describes. Primarily, the *about location* is estimated based on geographical references (e.g., gazetted terms) in tweets. Geographical references can be missing, ambiguous, and of different granularities. Consequently, they cause challenges in determining *about locations*. For instance, while *Effiel tower* in Example (2.52) suggests a unique fine-grained location in Paris, France, *Bay area* and *California* in Example (2.53) refers to a coarse-grained area that involves multiple cities, and thus the city-level *about location* in Example (2.53) is ambiguous.

The *tweeting location* is the location where a tweet is actually sent from. It is always unique and its granularity also ranges from precise points to cities and countries. While a tweet may contain geographical references to infer the *about location*, its *tweeting location* is usually unknown, unless a reliable source of geospatial information is provided such as a GPS label. GPS-labelled tweets contain precise latitude and longitude coordinates where they are sent from. Example (2.54) is such a tweet with coordinates $(-37.80, 144.96)$ embedded in the tweet metadata.

(2.54) DMD building gets a fire alarm again. #goodExecuseForNotWritingTheisis

Finally, a user's *primary location* denotes the location where the user is primarily based. Given that Twitter users may move around within a geographical region, the granularity of *primary location* is often coarse-grained, e.g., cities. A Twitter user may stay in multiple locations (Li *et al.* 2012a), especially for frequent travellers. Nevertheless, frequent travellers do not account for a large proportion of Twitter users (Abrol *et al.* 2012). Compared with the previous two types of location, the *primary location* is derived from a user's aggregated tweets, based on the assumption that word choice and topics of conversation mentioned by a user are strongly influenced by the user's primary location. Tweets from Example (2.55) to Example (2.57) are excerpts from a Twitter user account. The user's *primary location* is Melbourne, Australia based on local knowledge of the city, e.g., *Melb*, *Docklands*, and *Queen @USER Market*. It would be labour-intensive to determine *primary locations* for Twitter users based on human judgement. In practise, the ground truth of *primary locations* is often determined by GPS labels in tweets (Eisenstein *et al.* 2010; Roller *et al.* 2012) or GPS labels in user metadata (Cheng *et al.* 2010).

(2.55) Congrats and yeeha! RT @USER: FT: AUS 16 - 15 LIO: We're heading to Sydney for the decider! #AUSvLIO

(2.56) We support cycling in Melb & #Ride2Work. Visit us tomorrow am at City Sq, Docklands, City Baths & Nth Melb Rec Centre....

(2.57) Queen @USER Market Week kicks off today, with lots of fab free foodie fun planned. URL

In many cases, these three locations are consistent with each other, e.g., it is reasonable to believe that a user's *primary location* is from a place p, if most tweets talk about p and many GPS-labelled tweets are from p. Sometimes, the locations are inconsistent, e.g., Sydney, Australia, is a possible *about location* in Example (2.55), different from Melbourne, Australia, the user's *primary location*.

Because the *primary location* is based on aggregated tweets which contain more geospatial information than *about locations* and *tweeting locations*, most existing geolocation work focuses on user-level *primary location* prediction (Cheng *et al.* 2010; Roller *et al.* 2012).

Challenges in Twitter Geolocation Prediction

Even though more geospatial information is obtained by aggregating a user's tweets, identifying their primary locations is non-trivial, mainly due to a lack of reliable geospatial information. Although Twitter allows users to declare their locations in user profiles, the location descriptions are often unstructured and ad hoc (Cheng *et al.* 2010; Hecht *et al.* 2011), e.g., people use vernacular expressions such as *philly*, or misspellings such as *Filladephia* to refer to *Philadelphia*; non-geographical descriptions such as *in your heart* are also commonly found. Without appropriate processing, the value of these location fields is greatly limited. Hecht *et al.* (2011) demonstrated that directly feeding these declared locations into off-the-shelf tools for geolocation prediction is ineffective. Alternatively, some tweets sent from mobile devices are geotagged with precise GPS coordinates, however, the proportion of geotagged tweets is estimated to be approximately 1-2% (Cheng *et al.* 2010; Priedhorsky *et al.* 2014) and the locations of the vast majority of users are not geotagged. Methods based on IP addresses (Buyukkokten *et al.* 1999) have limited utility in the context of social media, where the IP address from which a user is sending messages is typically not known (other than to the service provider). Moreover, geographical divisions of IP addresses are not always credible. For instance, branches of an international corporation might use the same IP address range, but their true locations could be spread across the world. In addition, virtual private

networks (VPNs) complicate things because a shown IP address might not reflect the true location of a user.

Because Twitter data does not contain adequate and reliable geospatial information for existing location prediction methods, alternative approaches have been proposed to utilise the noisy but abundant geospatial information. The following sections discuss such approaches to Twitter geolocation prediction, including social network-based approaches and text-based approaches. Specifically, network-based methods make use of online social relationships and text-based methods mainly exploit geospatial references (e.g., gazetted terms, and dialectal words) in tweets.

2.3.2 Network-based Geolocation Prediction

Geographical proximity of social ties has been observed in many real-life social networks (Wellman 1979; Wellman *et al.* 1988), that is, for a given user u, most of u's friends' locations will be geographically close to u, because it is easier to form social relationships with local people than with people who live far away. Empirical studies (Gruzd *et al.* 2011) have shown that many social media users take their real-life social ties online. As such, it is reasonable to assume that many online social friends of u have geographical locations close to u.[24] This geographical proximity assumption for online users has been observed in many social networks such as Facebook (Backstrom *et al.* 2010). Based on these observations, the geolocation of a Twitter user could be estimated by examining the locations of the user's social relationships. Both explicit friendships (Sadilek *et al.* 2012a; Rout *et al.* 2013) and implicit social interactions (Chandra *et al.* 2011; Jurgens 2013), have been shown to be effective in predicting locations of social media users.

Abrol *et al.* (2012) assumed the location of a Twitter user is strongly influenced by the community that the user belongs to. Therefore, they identified social clusters in which all users in the cluster are mutually connected. The location of a user is

[24] Geolocation is not the only factor that shapes the online social relationships. Factors such as communities and languages also affect the formation of these relationships (Takhteyev *et al.* 2012). For instance, NLP researchers may become online friends even their primary locations are far away from each other.

represented by the location of the social cluster which is further determined by the majority location of its members. However, a user may appear in many social clusters such as colleagues, friends and family members, ending up with multiple derived locations. For such cases, the final location is obtained by the majority location from all social clusters.

Rout *et al.* (2013) applied an SVM classifier to incorporate location priors and subgraph structures in the social network to predict locations. For instance, they analysed the city population as a prior in predictions, and the cities were further binned into buckets based on population, forming features in the SVM classifier. They further examined the impact of reciprocal friendships and co-friendships (i.e., users sharing common friends) on location prediction accuracy. The experimental results show that population density and reciprocal friendship are influential factors contributing to social network-based location prediction.

Despite their simplicity, majority vote-based approaches have been shown to be effective on various datasets, locating Twitter users within just a few kilometres of their primary locations (Rout *et al.* 2013). However, such promising results are challenged by some pragmatic issues. The first one is efficiently obtaining reliable social network information, e.g., reciprocal following relationships. It is non-trivial to fully reconstruct and maintain a dynamic relationship graph for social media sites such as Twitter, and the rate limiting of the Twitter API further hampers any such effort. Beyond the technical obstacles, challenges also come from the nature of the data. Isolated users who are unwilling to follow or to be followed (e.g., a private account for tweeting personal updates) would challenge the assumption of social network-based methods (Davis Jr. *et al.* 2011). Moreover, network-based approaches also depend on the availability of friends' locations. In practise, many Twitter users have neither geotagged tweets nor canonical unambiguous locations (Cheng *et al.* 2010; Hecht *et al.* 2011). As a result, a user's friends' locations are largely unknown, which results in a chicken and egg problem for social network-based methods.

To compensate for the low ratio of friends' locations, Abrol and Khan (2010) took a cascaded inference strategy. Given an unknown user u, if u's friends' locations are unknown (i.e., those users have no geotagged tweets, and leave their metadata

fields blank), then the friends' locations are further estimated by friends' friends' locations, recursively. As such, u's location is estimated by propagating location information from the "closest" friends with a known location. Similarly, Jurgens (2013) bootstrapped from a small number of seed users whose locations are reliably known through, for example, geotagged tweets. Then, he extended a semi-supervised label propagation algorithm to iteratively infer an unknown user's location based on the locations of the user's friends. Additionally, he adopted symmetric tweet user mentions to construct the social network, rather than using reciprocal following relationships, which makes it easier to construct the social relationship graph and is not rate limited by Twitter. Having discussed the challenging issues of social network-based methods, the next section will address text-based geolocation predictions that complement network-based methods.

2.3.3 Text-based Geolocation Prediction

The fundamental idea of text-based geolocation prediction is to utilise geospatial references in tweet text to infer locations. These references range from gazetted terms (e.g., *Melbourne*) and local entities (e.g., *Yarra river*), to dialectal and regional words (e.g., *Aussie* and *footy*).[25] It is reasonable to assume that user posts in social media reflect their geospatial locum, because word priors and topics discussed in Twitter differ from region to region.

Off-the-shelf Methods

Various types of off-the-shelf geolocation services have been developed in industry and academia. Gazetted terms like city names are exploited to infer geolocations. Intuitively, if a place name is frequently mentioned by a user in their tweets, it is likely the user is from that region. Geolocation services are distinguished based on their functions (Leidner 2007), and some common services are as follows:

- *Geoparsing* extracts geospatial references from unstructured text, e.g., *Brunswick street* is obtained from *There's some very strange people on Brunswick street.*

[25]These examples are based in Melbourne, Australia.

The primary challenge of geoparsing is ambiguity, including determining whether an expression in a text is a geospatial term or not (Leidner and Lieberman 2011).

- *Geocoding* maps structured textual addresses to explicit geo-coordinates (Leidner 2007), e.g., *800 Swanston Street, Carlton* is mapped to $(-37.80, 144.96)$ using the Google Maps API.[26] Geocoding services usually target unambiguous and complete addresses for precise location mappings.

- *Toponym identification and resolution* connects geoparsing and geocoding. It deals with uncertainty and incompleteness in place name resolution. This end-to-end solution first identifies ambiguous place names, and then resolves them to explicit geo-coordinates.

Popular geolocation services include the Google Maps API, Yahoo! PlaceFinder and PlaceSpotter,[27] MetaCarta,[28] and Unlock Text.[29] The Google Maps API primarily accepts well-formed short text addresses for geocoding. In contrast, Yahoo! PlaceSpotter accepts longer plain text for place term extraction and then disambiguates the toponyms to a unique location. It is not clear what the underlying methodologies used in these services are, however, preliminary results showed that applying off-the-shelf geolocation services to Twitter data is ineffective (Kinsella *et al.* 2011; Hecht *et al.* 2011; Graham *et al.* 2013).

In addition to off-the-shelf geolocation services, many methods adapt existing tools and resources in geolocation prediction. These methods range from naive place name matching and rule-based approaches (Bilhaut *et al.* 2003), to machine learning-based methods (primarily based on recognising named entities: Quercini *et al.* (2010); Gelernter and Mushegian (2011)). Despite the encouraging results of this approach on longer and more homogeneous document sets (Quercini *et al.* 2010), its performance is impeded by the nature of tweets: they are short and informal, and the chances of a user not mentioning gazetted places in their tweets is high. Moreover, the handling

[26] `https://developers.google.com/maps/documentation/geocoding/#Geocoding`

[27] `http://developer.yahoo.com/boss/geo/`

[28] `http://www.metacarta.com/`

[29] `http://edina.ac.uk/unlock/texts/`

of vernacular place names, e.g., *mel* for *Melbourne*, in this approach is limited. The reliance on named entity recognition is thwarted by the unedited nature of social media data, where spelling and capitalisation are much more ad hoc than in edited document collections (Ritter *et al.* 2011).

Recently, Dredze *et al.* (2013) proposed an efficient way to produce reliable geolocation prediction based on shallow Twitter text and metadata processing. Twitter place entities (i.e., entities incorporating user-tagged locations for tweets), user-declared profile locations and GPS coordinates are utilised in a cascaded way. Specifically, they matched these location-referenced fields against external gazetteers, and resolved GPS coordinates to cities, states and countries. The system yields reliable predictions by leveraging existing geographical resources embedded in tweets, however it comes at the cost of low user coverage. Many Twitter users do not send tweets with GPS coordinates or put their accurate physical locations in their profiles (Cheng *et al.* 2010).

Twitter data has limited geolocation information that is highly reliable and unambiguous. The unedited nature of Twitter data means it is different from the training data in existing geolocation services, and consequently these geolocation services perform poorly on Twitter data. In contrast, the geospatial references embedded in tweet text are not yet fully exploited in geolocation prediction. Recent research has therefore turned to more advanced methods leveraging such information.

Language Model-based Methods

Moving beyond off-the-shelf methods that primarily depend on gazetted terms, many robust machine learning methods model textual content (i.e., tweets) for geolocation prediction. For example, a user in London, UK is much more likely to talk about *BBC* and *tube* in tweets than a user in New York, US or Beijing, China. That is not to say that those terms are geographical references uniquely associated with London, of course: *tube* could certainly be mentioned by a user outside of the UK. However, the use of a range of such terms with high relative frequency is strongly indicative of the fact that a user is located in London. Following this intuition, many

approaches estimate locations based on the "bag-of-words" in tweets.

In general, two fundamental types of probabilistic models are primarily used in geolocation prediction model learning and inference (Ng and Jordan 2002). Generative models (e.g., naive Bayes) are based on estimation of the location class priors (e.g., $P(c_i)$, where c_i represents the ith location) and the probability of observing a given term vector given a location class (i.e., $P(w_1, w_2, \ldots, w_n|c_i)$, where $w_1, w_2, \ldots$ are terms generated from c_i). In contrast, discriminative models estimate the probability of a location class given a term vector (i.e., $P(c|w_1, w_2, \ldots, w_n)$).

Specifically, Wing and Baldridge (2011) divided the world's surface into uniform-sized grid cells, and compared the distribution of words in a given user's tweets to those in each grid cell using Kullback–Leibler (KL) divergence to identify that user's most likely location. One limitation of this approach is that grid cells in rural areas tend to contain very few tweets, while there are many tweets from more urban grid cells. Roller *et al.* (2012) therefore extended this method to use an adaptive grid representation based on a k-d tree (Bentley 1975). The adaptive grid cells cover the whole world's surface, but vary in size. Each grid cell is a near-rectangular polygon and contains approximately the same amount of data. Locations are represented using grid centroids, and the prediction is also based on the KL divergence between a user's tweets and the tweet data in each grid cell.

Kinsella *et al.* (2011) calibrated geolocation prediction at different granularities (e.g., zip code, city, state and country) and levels (e.g., tweet- and user-level). They compared various generative and discriminative models to benchmark the geolocation accuracy. The results revealed the intrinsic difficulty of geolocation prediction. Furthermore, the results also re-confirmed that simple geolocation models learnt from Twitter data outperform off-the-shelf tools.

Chandra *et al.* (2011) grouped the tweet replies in a conversation with the tweets of the first tweet author (whose location is known through GPS labels), as a means of obtaining extra geospatial information. The clustered tweets are used to estimate a per-word city distribution in a discriminative model. The prediction is obtained from the aggregated city probabilities from words in a test user's tweets.[30]

[30]The authors exploited implicit social interactions, however, the geolocation prediction is on the

Modelling distributions using "bag-of-words" in location prediction requires additional refinements, because a large number of words without any geospatial information exist in tweets, and these words may mislead the geolocation prediction model. Therefore, instead of geolocating users based on the whole textual data, priority is given to words that are indicative of location, e.g., city names, local entities, and dialectal words. Through this lens, various feature selection methods have been proposed to obtain these words. Next, we briefly review representative feature selection methods used in social media for selecting words carrying geospatial information. More details on feature selection literature and methods are presented in Section 4.3.

Cheng *et al.* (2010) developed a supervised binary SVM classifier to determine local words based on word frequency and distribution shape (Backstrom *et al.* 2008). Preference is given to words that have peaky geographical distributions, i.e., a local word's probability distribution is strongly skewed to the centre of a location, but then quickly drops off moving away from that location. The selected local words are then used to train a simple generative model for geolocating users from the continental United States. Additionally, they compensated for data sparsity using various smoothing techniques to further improve the geolocation prediction accuracy.

Different to Cheng *et al.*'s (2010) supervised approach, recent methods have been proposed to perform feature selection without supervision. Chang *et al.* (2012) pruned noisy data based on geometrically-local words (i.e., words whose usages occur geographically close to each other, and that are only found in a small number of cities), and words that are dis-similar to stop words in terms of the distribution of cities in which they are found. They experimented with the resultant feature set using both Gaussian mixture models (GMMs) and Maximum Likelihood Estimation (MLE), and achieved better prediction results than models trained without feature selection.

Ren *et al.* (2012) applied inverse location frequency to select words occurring in only a few places. They also set a maximum threshold on the average distance between pairwise locations in which a word occurs. The higher-ranked words then have the "local" property that they occur in only a few geographically close locations.[31]

basis of tweet text. Therefore, this approach is categorised as a text-based method.

[31]Their approach is a hybrid method combining both text-based and social network-based models.

Laere *et al.* (2013b) proposed an even more diverse set of feature selection methods. For instance, statistical hypothesis tests such as χ^2 are applied to estimate the strength of correlation between a word and a location. Furthermore, heuristic metrics are devised to capture the geographical spread of a word's usage, similar to Chang *et al.*'s (2012) method. In addition, they also considered spatial statistics in feature selection such as Ripley's K function (O'Sullivan and Unwin 2010), which is a randomised method that prefers words with lower average distances between pairwise locations, similar to Ren *et al.*'s (2012) method. Although this work is primarily evaluated on Flickr image tags, the methods apply equally to Twitter text.

Instead of selecting a subset of features (e.g., city names, dialectal words), Priedhorsky *et al.* (2014) proposed to include all words and assign weights to these words accordingly. Specifically, they applied a GMM to estimate the densities of n-grams (from tweet text to some geospatial metadata fields), which is analogous to modelling per n-gram location distributions. Consequently, common words like *today* exhibit a more flat distribution, while local words like *Washington* are more skewed in the GMM model.

Enhanced Language Model-based Methods

Beyond pure language model-based methods, other sources of textual information have also been integrated to enhance the accuracy of geolocation prediction.

Li *et al.* (2011) explored text and temporal factors in grounding a group of tweets to places-of-interest (POIs) — a pre-defined set of locations. Taking a government building and a cinema for example, the words used in tweets and the peak tweeting times in these two places are different. They used KL divergence in a language model-based method to compare tweet data differences. For the temporal factor, they examined daily, weekly, and monthly tweeting probabilities from a POI. These probabilities are linearly integrated with the language model-based scores in prediction.

Mahmud *et al.* (2012) considered tweet content (including hashtags and place names), location-based service (LBS) history and timezone information in geolocation

Here we only discuss their feature selection component.

prediction. They trained multinomial Bayes classifiers for each source and combined them in ensemble learning. Furthermore, they experimented with a decision tree model, in which a timezone-based classifier is first applied, then the tweet text and LBS-based classifiers are combined to predict locations within the timezone.

Gonzalez *et al.* (2012) applied a cascaded geolocation resolution framework based on tweet data, external gazetteers, and common rule patterns. The location is inferred in a coarse- to fine-grained manner, i.e., from a country to a city. In addition, the predictions incorporate temporal factors, accommodating for users moving around and having different locations at different times.

Schulz *et al.* (2013) combined scores from heterogeneous data sources including tweet text and metadata in user profiles. Scores derived from each source of geospatial information are summed up, and scaled to "aggregated height" on a polygon-partitioned map. Each polygon represents a location and the highest polygon is the prediction.

Topic Modelling-based Methods

Topics discussed on Twitter also vary across geographical regions. Intuitively, for instance, Americans are more likely to talk about *NBA* and *baseball* than Australians (who probably mention *AFL* and *rugby* more often). To capture these regional differences in topics, topic modelling-based approaches have been used to incorporate geographical factors in the generative process. The approach has been applied in similar geolocation prediction tasks such as Flickr image geotagging (Yin *et al.* 2011).

Recently, Eisenstein *et al.* (2010) proposed a topic modelling approach which incorporates a geographical variable (r). Instead of generating an observed word w from a per-word topic distribution ϕ_z as in the standard Latent Dirichlet Allocation (LDA) model (Blei *et al.* 2003), their proposed approach refines this step by additionally modelling the topic distributions across different geographical regions, i.e., w is generated from a per-word region-topic distribution ϕ_{rz}. Therefore, the observed user locations are generated from geographical regions and the region variable in topic modelling is linked with user locations. Generally, user locations are predicted at the

regional level by adopting the location centroid for geotagged tweets from that region.

Hong *et al.* (2012) further improved the approach by considering more fine-grained factors in an additive generative model. In addition to introducing per-region topic variance, they incorporated per-user topic variance, a regional language model, and global background topics. To compensate for the computational complexity associated with these extra hidden variables, they adopted sparse modelling in inference.

Ahmed *et al.* (2013) clustered flat representations into a hierarchical representation for both tweet content and locations. The basic idea is that hierarchical structure captures the relative "closeness" between classes better than flat structure. For instance, tweets from Portland, US are more similar to tweets from Seattle, US, but are much less similar to tweets from Singapore. They incorporated this intuition in a nested Chinese Restaurant Franchise and achieved better results than the flat version.

Despite the benefits of incorporating per-region topic variance in these models, a weakness of topic modelling-based approaches is their efficiency. It is generally computationally expensive to model topics for large volumes of data, such as that available through social media. In contrast, language model-based approaches are more practical and attractive among text-based geolocation prediction methods, because they are more efficient when training geolocation models.

Similar to network-based approaches that fail to geolocate users without any social relationships, text-based methods are also incapable of geolocating users who only follow other users but never tweet. In addition, text-based methods are less effective if users do not discuss local topics or do not use dialectal words, either because they are aware of potential privacy leaks when tweeting or because of their personal tweeting style.

2.3.4 Hybrid Methods

Having discussed text-based and social network-based approaches, we now discuss methods that combine the two approaches together for better accuracy.

Abrol and Khan (2010) estimated an unknown user u's location based on u's friends' locations. The locations of u's friends are estimated using gazetted terms in

their tweets, and are further represented in the form of location distributions. This is because there may be ambiguous gazetted terms in their tweets, and a friend may mention many places in tweets. An example distribution of a friend's location is 80% in Melbourne, Australia, 15% in Sydney, Australia and 5% in other cities. In the prediction stage, u's friends' location distributions are summed, and the final prediction is then the most likely location in this distribution.

Ren *et al.* (2012) built a generative model that makes use of local words and named entities as the text-based part of the geolocation model. In the social network-based part, they combined three sources of social relationship in majority vote for a user u: (1) locations of u's followers; (2) locations of users that u is following; and (3) the followers of u's *siblings* — suppose s is following a user that u is also following, then the follower locations of s are also counted for u. Both text-based and social network-based prediction scores are scaled to $[0, 1]$, and are combined linearly to select the final prediction.

Sadilek *et al.* (2012a) jointly predicted social relationships and user locations in Twitter. They made use of co-friendships (i.e., two users who share the same friend), word choice and temporal activity overlap in a Bayesian propagation framework. The model recovers hidden social relationships and user locations based on partially observed data. Although promising results have been achieved, the approach requires users to be actively posting GPS-labelled tweets, limiting its applicability to densely populated areas and users with more tweet data. Furthermore, their results suggest co-friendships are effective in locating users, different from Rout *et al.*'s (2013) finding. One potential reason is because of the different data used in the studies. Sadilek *et al.* (2012a) primarily focused on active users with at least 100 geotagged posts per month in big cities. These active users often have a higher ratio of social connectivity, and consequently the social graph is relatively dense. The co-friendships in Rout *et al.* (2013), on the other hand, are country-wide (i.e., within UK), and also incorporate less-active users who only tweet once a month.

Similarly, Li *et al.* (2012b) jointly combined user tweet data and social relationships in a directed graphical model. They considered both users and locations as nodes, and these nodes are connected by two types of edges which represent: (1) a

user tweeting about a place; and (2) a user following another user, corresponding to the text-based part and social network-based part, respectively, in their model. All location nodes themselves are associated with geolocations, and the user nodes are partially observed (i.e., some users have canonical unambiguous locations). The social network-based inference then propagates the location information from observed nodes to unobserved user nodes (i.e., users whose locations are not known). As for the text-based part, a user's location is estimated based on geolocation references in their tweets such as gazetted terms. Their experiments suggest optimising location predictions over the whole graph outperforms inferring a user's location based on nearby nodes.

As a probabilistic generalisation of Li *et al.*'s (2012b) method, Li *et al.*'s (2012a) method allows users to have multiple locations in their model. A user might tweet about a location if they are there, and the user's friends may stay in multiple places. As such, they assume a user has a primary location and some temporary locations forming a multinomial distribution over locations. The goal is to estimate the location distribution for the user based on partially observed data, i.e., some users with known primary locations. They incorporated these intuitions in a generative process in LDA. The tweeting and following edges are generated based on: (1) the background random model, and (2) the location assigned from the user's multinomial location distributions.

2.3.5 Summary

In this section, we discussed the benefits and challenges of geolocation awareness in social media. We further categorised geolocations by granularity and location type. After that, we reviewed mainstream approaches to geolocation prediction. Off-the-shelf tools are often ineffective due to non-standard and ambiguous geographical references in social media text. Most existing work has moved to geolocation prediction using less reliable but more abundant information. For instance, social network-based methods predict a user's location based on the user's social relationships (e.g., friends' locations), and text-based methods rely on geospatial references

(e.g., gazetted terms, dialectal words, local topics) embedded in the text to disambiguate the locations. Combining all these methods improves geolocation prediction, however, the integration of different approaches also increases the computational burden, which is a non-trivial factor when processing high volumes of social media data. To balance efficiency and effectiveness of geolocation prediction, both features and learning algorithms require careful selection.

In this thesis, we exclusively focus on improving text-based methods for geolocation prediction. In particular, we extend the reach of existing text-based methods, and examine a range of influential factors on prediction accuracy in Chapter 4, such as a more detailed exploration of feature selection methods (in Section 4.3) and the impact of user tweeting language. Making sense of the impact of these factors is crucial, because they are often mutually influencing, and may drastically change the prediction accuracy in practise.

2.4 Literature Summary

In this chapter, we described the impact and characteristics of social media. In particular, we discussed the nuts and bolts of Twitter data. Twitter data is noisy in content and massive in volumes, challenging existing NLP tools in both accuracy and efficiency. This thesis aims to improve the effectiveness and efficiency of social media data for NLP tasks and applications. It concentrates on two Twitter processing tasks: text normalisation and geolocation prediction. We summarised the related work for each task and identified gaps between Twitter data and the existing methods. In the next two chapters, we move on to our work on text normalisation (Chapter 3) and user geolocation (Chapter 4).

Chapter 3

Text Normalisation

In this chapter, we explore and evaluate Twitter text normalisation approaches. First, the English lexical normalisation task is defined in Section 3.1. After that we perform a pilot study on tweet samples which motivates the development of a token-based normalisation approach in Section 3.2. In addition, we also compare the developed method with existing benchmarks, and conduct a preliminary investigation on the detection of OOV non-standard words in tweets. Inspired by the analysis on the token-based method and existing approaches, a more practical type-based approach is developed and evaluated in Section 3.3. This type-based method is further evaluated using a downstream POS tagging task in Section 3.4. Additionally, the effectiveness of the type-based approach is demonstrated on a Spanish text normalisation task in Section 3.5. After that, we discuss recent progress on Twitter text normalisation in Section 3.6. Because many new methods have been proposed since the work described in this chapter, this discussion contrasts our work with recent literature and provides insights for the future development of text normalisation methods. Finally, we summarise the chapter in Section 3.7.

3.1 Normalisation Scope

Following Figure 2.1 in Section 2.2.2, we define the lexical normalisation task as a mapping from non-standard words to their standard In-Vocabulary (IV) surface

forms with two extra restrictions:

- Only Out-Of-Vocabulary (OOV) words are considered for normalisation;
- Normalisation is restricted to a single-token word.[1]

An immediate implication of the task definition is that non-standard words which happen to coincide with an IV word (e.g., *can't* spelled as *cant*) are outside the scope of lexical text normalisation. Furthermore, deabbreviation of acronyms and initialisms (e.g., *imo* "in my opinion") also largely fall outside the scope. Note that single-word abbreviations such as *govt* "government" are very much within the scope, as they are lexical variants and correspond to a single token in their standard lexical form. To make the boundary clear, we hereafter use "lexical variants" (Gouws *et al.* 2011a) to denote the target OOV non-standard words in this chapter.

Given this task setting, a necessary preprocessing step for normalisation is the identification of lexical variants for normalisation. All tokens that consist of alphanumeric characters are examined. They are categorised into IVs and OOVs relative to a dictionary, and only OOVs are eligible for normalisation. However, the OOVs include lexical variants, but also include other word types, such as neologisms and proper nouns, which happen to not be listed in the dictionary being used. One challenge for lexical normalisation is therefore to distinguish between the standard OOVs that should not be normalised (such as *hopeable* and *WikiLeaks*, which are not included in the dictionary we use in our experiments) and lexical variants requiring normalisation such as typos (e.g., *earthquak* "earthquake"), register-specific single-word abbreviations (e.g., *lv* "love"), and phonetic substitutions (e.g., *2morrow* "tomorrow"). Note that many previous normalisation approaches (Choudhury *et al.* 2007; Cook and Stevenson 2009) have made the assumption that lexical variants have already been identified. In the following sections, we begin by assuming lexical variants to be known in text normalisation. The issue of identifying lexical variants from amongst OOVs is addressed in Section 3.2.6.

[1]We set a single-token restriction because it is more tractable to evaluate the impact of normalisation (Eisenstein 2013b), and as our later pilot study in Table 3.1 shows, most non-standard words are single tokens.

Throughout this chapter, the `Aspell` dictionary is used to determine whether a token is OOV.[2] Furthermore, Twitter user mentions (e.g., *@twitter*), hashtags (e.g., *#twitter*) and URLs (e.g., *twitter.com*) are excluded from consideration for normalisation, but left in situ for future downstream processing. The language filtering of Twitter to automatically identify English tweets was based on `langid-2010`, using the EuroGov dataset as training data, a skew divergence nearest prototype classifier, and a mixed unigram/bigram/trigram byte feature representation (Baldwin and Lui 2010).[3]

Twitter text attracts users from diverse language backgrounds, and is therefore highly multilingual. The preliminary study on tweet samples shows that English data accounts for around half of the overall data (Hong *et al.* 2011). This chapter mainly focuses on English text normalisation, leaving the more general task of multilingual normalisation for future work.

3.2 Token-based Lexical Normalisation

3.2.1 A Pilot Study on OOV Words

To get a sense of the relative need for lexical normalisation, we perform an analysis of the distribution of OOV words in different text types. In particular, we calculate the proportion of OOV tokens per message (or sentence, in the case of edited text), bin the messages according to OOV token proportion, and plot the probability mass contained in each bin for a given text type. The three corpora we compare are the New York Times (`NYT`),[4] SMS,[5] and Twitter.[6] The results are presented in Figure 3.1. Both SMS and Twitter have a relatively flat distribution. For instance, Twitter has

[2]The dictionary is used by Aspell (v6.06) in Ubuntu 10.04. The file is located in `/usr/share/dict/american-english`. We remove all one character tokens, except *a* and *I*, and treat *RT* as an IV word. We chose this dictionary because it is widely used in many UNIX/Linux distributions.

[3]`langid-2010` tokenises a tweet into byte *n*-grams, so it isn't sensitised to individual tokens.

[4]Based on 44 million sentences from English `Gigaword` (David Graff 2003)

[5]Based on 12.6 thousand SMS messages from How and Kan (2005) and Choudhury *et al.* (2007).

[6]Based on 1.37 million tweets collected from the Twitter streaming API from August to October 2010, and filtered for monolingual English messages using `langid-2010`.

a long tail: around 15% of tweets have 50% or more OOV tokens.[7] It suggests many OOV words in SMS and Twitter co-occur in one message, and this makes context modelling difficult. In contrast, `NYT` shows a power law with long-tail word frequency distribution, despite the large number of proper nouns it contains. Recent research (Hu *et al.* 2013) also confirms these relative OOV token ratios among different sources of text.

While this analysis suggests that Twitter and SMS are similar in being heavily laden with OOV tokens, it does not shed any light on the relative similarity in the makeup of OOV tokens in each case. To further analyse the two data sources, we extracted two lists of OOV terms — those found exclusively in SMS, and those found only in Twitter — and sorted each list by frequency. Manual analysis of high-frequency items in each list revealed that OOV words found only in SMS were largely personal names (e.g., *shuhui*, *yijue*), while the Twitter-specific set, on the other hand, contained a more heterogeneous collection of OOVs including: proper nouns (e.g., *bieber*), lexical variants (e.g., *smh* interpreted as "shake my head" or "somehow") and non-English words (e.g., *que* is a Spanish word). Despite the different volumes of these datasets, this finding suggests that Twitter is a noisier data source in terms of OOV types, and hence that text normalisation for Twitter needs to be more nuanced than for SMS.

To further analyse the lexical variants in Twitter, we randomly selected 449 tweets and manually analysed the sources of variation, to determine the phenomena that lexical normalisation needs to deal with. We identified 254 token instances of lexical variants, and broke them down into categories, as listed in Table 3.1. "Letter" refers to instances where letters are missing, permuted, or redundant, but the lexical correspondence to the target word form is accessible via letter manipulations (e.g., *shuld* "should"). "Number Substitution" refers to instances of letter–number substitution, where numbers have been substituted for phonetically-similar sequences of letters (e.g., *4* "for"). "Letter&Number" refers to instances which have both letter variations and number substitutions (e.g., *b4* "before"). "Slang" refers to instances

[7]This number is obtained by summing up the y-values for points corresponding to $> 50\%$ OOV tokens per message (i.e., the right half of the figure).

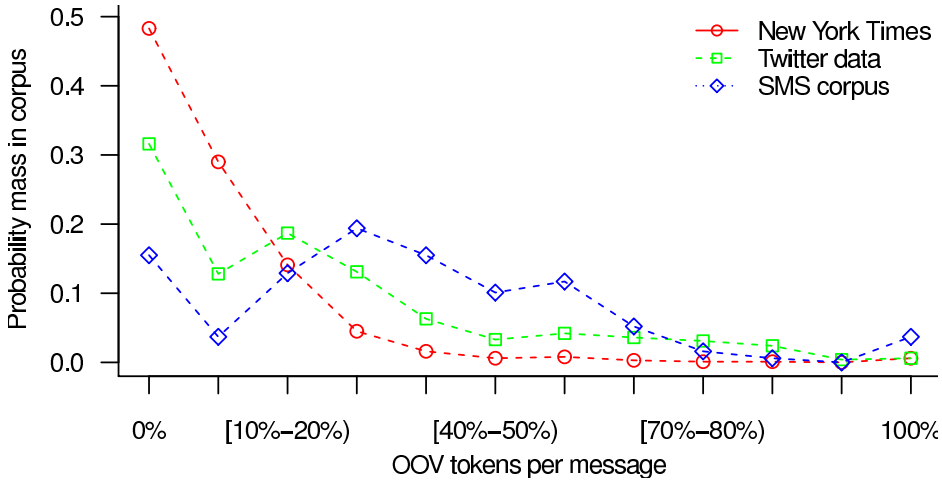


Figure 3.1: Out-of-vocabulary word distribution in English `Gigaword` (`NYT`), Twitter and SMS data.

Category	Ratio
Letter&Number	2.36%
Letter	72.44%
Number Substitution	2.76%
Slang	12.20%
Other	10.24%

Table 3.1: Categorisation of lexical variants.

of Internet slang (e.g., *lol* "laugh out loud"), as found in a slang dictionary (see Section 3.2.4). "Other" is the remainder of the instances, which is predominantly made up of occurrences of spaces having been deleted between words (e.g., *sucha* "such a").[8] If a given instance belongs to multiple error categories (e.g., "Letter&Number" and it is also found in a slang dictionary), we classify it into the higher-occurring category in Table 3.1.

Acknowledging other categorisation methods of lexical variant formations (Thurlow 2003; Cook and Stevenson 2009), our classification is shaped to coordinate the

[8]We don't touch these concatenated words, in accordance with our task definition.

downstream normalisation. As discussed in Section 2.2.1, it is often difficult to track the exact causes of lexical variants, but this coarse-grained categorisation shed lights on the general formation of lexical variants. From Table 3.1, it is clear that "Letter" accounts for the majority of lexical variants in Twitter, and that most variants are based on morphophonemic variations. This empirical finding assists in shaping our strategy for lexical normalisation.

3.2.2 Datasets and Evaluation Metrics

The aim of our experiments is to compare the effectiveness of different methodologies over short messages in social media. We present evaluations on two datasets, including: (1) a SMS corpus (Choudhury *et al.* 2007) for benchmarking; and (2) a novel Twitter dataset developed as part of this research, based on a random sampling of 549 English tweets. The English tweets were annotated by three independent annotators with NLP backgrounds. All OOV words were automatically pre-identified, and the annotators were requested to determine: (a) whether each OOV word was a lexical variant or not; and (b) in the case of tokens judged as lexical variants, what the standard form was, subject to the task definition outlined in Section 3.1. The total number of lexical variants contained in the SMS and Twitter datasets were 3849 and 1184, respectively.[9]

As discussed in Section 3.1, much previous work on SMS data has assumed perfect lexical variant detection and focused only on the identification of standard forms. Here we also assume perfect detection of lexical variants in order to compare our proposed approach to previous methods. We consider token-level precision, recall and F-score ($\beta = 1$), and also evaluate using BLEU (Papineni *et al.* 2002) over the normalised form of each message. We consider the latter measure mainly because statistical machine translation-based (SMT) approaches to normalisation (which we compare our proposed method against) can lead to perturbations of the token stream, vexing evaluation using standard precision, recall and F-score.

[9]The Twitter dataset is available at `http://www.csse.unimelb.edu.au/research/lt/resources/lexnorm/`

$$P = \frac{\text{\# correctly normalised tokens}}{\text{\# normalised tokens}}$$
$$R = \frac{\text{\# correctly normalised tokens}}{\text{\# tokens requiring normalisation}}$$
$$F = \frac{2PR}{P+R}$$

3.2.3 Token-based Normalisation Approach

Having explored lexical variants in Section 3.2.1, this section proposes a generation-and-selection lexical normalisation strategy involving: (1) confusion set generation, in which we identify IV normalisation candidates for a given lexical variant; (2) candidate selection, where we select the best standard form of the given lexical variant from the candidates generated in (1).

Confusion Set Generation

In generating possible normalisation candidates, the following steps are utilised. First, inspired by Kaufmann and Kalita (2010), any repetitions of more than 3 letters are reduced back to 3 letters (e.g., *cooool* is reduced to *coool*). Second, IV words within a threshold of T_c in terms of character edit distance of a given OOV word are considered, a heuristic widely used in spell checkers. Third, the Double Metaphone algorithm (Philips 2000) is used to decode the pronunciation of all IV words; IV words within an edit distance of T_p of a given OOV word, under phonemic transcription, are also included in the confusion set. This allows us to capture OOV words such as *earthquick* "earthquake". In Table 3.2, we list the recall and average size of the confusion set generated by the final two strategies with different threshold settings, based on our evaluation dataset (see Section 3.2.2).

The recall for lexical edit distance with $T_c \leq 2$ is moderately high, but it is unable to detect the correct candidate for about one quarter of words. The combination of

Algorithm 1: Confusion Set Generation

Input: An OOV word (oov), edit distance thresholds for characters (T_c) and phonemic codes (T_p), a dictionary of IV words ($DICT$), and the proportion of candidates to retain after ranking by language model (R_{lm})

Output: Confusion set for OOV word (C_{set})

oov = RemoveRepetitions(oov);
$C_{set} \leftarrow \{\}$;
forall the $iv \in DICT$ **do**
  if *CharacterEditDistance(oov, iv)* $\leq T_c$ *or PhonemicEditDistance(oov, iv)* $\leq T_p$ **then**
    $C_{set} \leftarrow C_{set} \cup \{iv\}$;
  end
end
C_{list} = RankByTrigamModelScoreDesc(C_{set});
num_{list} = GetLength(C_{list}) $*R_{lm}$;
$index = 0$;
$C_{set} \leftarrow \{\}$;
repeat
  $C_{set} \leftarrow C_{set} \cup \{C_{list}[index]\}$
  $index{+}{+}$
until $index \geq num_{list}$;
return C_{set};

the lexical and phonemic strategies with $T_c \leq 2 \vee T_p \leq 2$ is more impressive, but the number of candidates has also soared. Note that increasing the edit distance further in both cases leads to an explosion in the average number of candidates, and causes significant computational overhead. Furthermore, a smaller confusion set is easier for the downstream candidate selection. Thankfully, $T_c \leq 2 \vee T_p \leq 1$ leads to an extra

Criterion	Recall	Average Candidates
$T_c \leq 1$	40.4%	24
$T_c \leq 2$	76.6%	240
$T_p = 0$	55.4%	65
$T_p \leq 1$	83.4%	1248
$T_p \leq 2$	91.0%	9694
$T_c \leq 2 \vee T_p \leq 1$	88.8%	1269
$T_c \leq 2 \vee T_p \leq 2$	92.7%	9515

Table 3.2: Recall and average number of candidates for different confusion set generation strategies.

increment in recall to 88.8%, with only a slight increase in the average number of candidates.

In addition to generating the confusion set, we further rank the candidates based on a trigram language model trained over 1.5GB of clean Twitter data (i.e., tweets which consist of all IV words) using `SRILM` (Stolcke 2002): despite the prevalence of OOV words in Twitter, the sheer volume of the data from Twitter Streaming API means that it is relatively easy to collect large amounts of all-IV messages relative to the `Aspell` dictionary. We truncate the ranking to the top 10% of candidates in our experiments, based on which the recall drops back to 84% with a 90% reduction in candidates. Based on these results, we use $T_c \leq 2 \vee T_p \leq 1$ with language model truncation as the basis for confusion set generation. The generation process is summarised in Algorithm 1.

Examples of lexical variants where we are unable to generate the standard forms are clippings such as *fav* "favourite" and *convo* "conversation".

Candidate Selection

We select the most likely candidate from the previously generated confusion set as the basis of normalisation. Both lexical string similarity and contextual information

is used, and the similarity scores are linearly combined in line with previous work (Wong *et al.* 2006; Cook and Stevenson 2009). As shown in Equation (3.1), each $f_i(w)$ represents a (string or contextual) similarity method of total n methods for candidate w, and the λ_i for each method is set to $1.0/n$.

$$score(w) = \sum_{i=1}^{n} \lambda_i f_i(w) \tag{3.1}$$

Lexical edit distance, phonemic edit distance, prefix substring, suffix substring, and the longest common subsequence (LCS) are exploited to capture morphophonemic similarity. Both lexical and phonemic edit distance (ED) are non-linearly transformed to $\frac{1}{exp(ED)}$ so that smaller numbers correspond to higher similarity, as with the subsequence-based methods.

The prefix and suffix features are intended to capture the fact that leading and trailing characters are frequently dropped from words, e.g., in cases such as *gainst* "against" and *talkin* "talking". We calculate the ratio of the LCS over the maximum string length between a lexical variant and a candidate, since the lexical variant can be either longer or shorter than (or the same size as) the standard form. For example, *mve* can represent either *me* or *move*, depending on context. We normalise these ratios so that the sum over candidates for each measure is 1, following Cook and Stevenson (2009).

For context inference, we employ language model-based features. Ranking by language model score is intuitively appealing for candidate selection, but our trigram model is trained only on clean Twitter data and lexical variants often don't have sufficient context for the language model to operate effectively, as in *bt* "but" in *say 2 sum1 bt nt gonna say* "say to someone but not going to say".

To consolidate the context modelling, we also obtain dependency features that are not restricted by contiguity. First, we use the Stanford parser (Klein and Manning 2003; De Marneffe *et al.* 2006) to extract dependencies from the `NYT` (see Section 3.2.1). For example, from a sentence such as *One obvious difference is the way they look*, we would extract dependencies such as `rcmod(way-6,look-8)` and `nsubj(look-8,they-7)`. We then transform the dependencies into simplified de-

pendency features, e.g., we would extract dependencies of the form `(look,way,+2)`, indicating that *look* occurs 2 words after *way*. We choose dependencies to represent context because they are an effective way of capturing positional relationships between words, and similar features can easily be extracted from tweets. Note that we don't record the dependency type here, because we have no intention of dependency parsing text messages, due to their noisiness and the volume of the data. The counts of dependency forms are combined together to derive a confidence score, and the scored dependencies are stored in a dependency bank.[10]

Although tweets consist of a mixture of genres, much more complex than the edited newswire text, we assume that words in the two data sources participate in similar dependencies based on the common goal of getting across the message effectively. The dependency features can be used in noisy contexts and are robust to the effects of other lexical variants, as they do not rely on contiguity. For example, *uz* "use" in *i did #tt uz me and yu*, dependencies can capture relationships like `aux(use-4,do-2)`, which is beyond the capabilities of the language model due to the hashtag being treated as a correct OOV word.

3.2.4 Baselines and Benchmarks

Baselines

We compare our proposed token-based normalisation approach to some off-the-shelf tools and simple methods. As the first baseline, we use the `Ispell` spell checker to correct lexical variants.[11] Furthermore, we set up a web-based language modelling approach to normalisation. For a given lexical variant, we first use the confusion set generation method (from Section 3.2.3) to identify plausible normalisation candidates. We then identify the lexical variant's left and right context tokens, and use the `Web 1T` 5-gram corpus (Brants and Franz 2006) to determine the most frequent 3-gram

[10]The confidence score is derived from the proportion of dependency tuples. For example, assume an OOV word `O` has two IV normalisation candidates `A` and `B`, and `CW` is a word in the context of `O`. `(A,CW,+1)` and `(B,CW,+1)` are the two corresponding dependency tuples, and occur 200 and 300 times respectively in the corpus. The confidence score for `A` and `B` would be calculated as 0.4 and 0.6, respectively.

[11]We use `Ispell` 3.1.20 with the `-w/-S` options to get the most probable correct word.

(one word to each of the left and right of the lexical variant) or 5-gram (two words to each of the left and right). Lexical normalisation takes the form of simply identifying the IV word of the highest-frequency n-gram which matches the left/right context, and where the word is in the lexical variant's candidate set. Finally, we also consider a simple dictionary lookup method using 5021 slang items collected from the Internet.[12] We substitute any usage of an OOV having an entry in the dictionary by its listed standard form.

Benchmarks

We further compare our proposed method against previous methods, which we take as benchmarks. We reimplemented a representative spell checking-based method (Cook and Stevenson 2009) and the SMT approach of Aw *et al.* (2006), which is widely used in SMS normalisation. The phrasal SMT benchmark was based on Moses (Koehn *et al.* 2007), with synthetic training and tuning data of 90,000 and 1000 sentence pairs, respectively. The clean data is randomly sampled from the 1.5GB of clean Twitter data, and the parallel noisy data with lexical variants is synthesised according to the error distribution of the SMS corpus.[13] The 10-fold cross-validated BLEU score over this data is 0.81.

3.2.5 Results Analysis and Discussion

In Table 3.3, we compare baselines and benchmarks with our combined method (DWC) which consists of dictionary lookup (DL), word similarity (WS) and context support (CS). The latter two models (as discussed in Section 3.2.3) are further linearly combined with equal weights (WC). In the combined method, a lexical variant is firstly substituted using DL, then the remaining variants are normalised in WC. Additionally, we also determine the relative effectiveness of the component methods of DWC.

[12] `http://www.noslang.com`

[13] Suppose the distributions of u and "you" in the SMS corpus are 30% and 70%, respectively, then the synthesised data is generated by replacing each "you" with u with a 30% likelihood in clean tweets.

From Table 3.3, we see that the general performance of our proposed method over Twitter is better than that over the SMS dataset. To better understand this, we examined the annotations in the SMS corpus, and found them to be less conservative than ours, due to the different task specification. In our annotations, the annotators were instructed to only normalise lexical variants if they were confident of how to normalise, as with *talkin* "talking". For lexical variants where they couldn't be certain of the standard form, the tokens were left untouched. However, in the SMS corpus, annotations such as *sammis* are mistakenly recognised as a variant of "same", but actually represent a person name. This leads to a performance drop for most methods over the SMS corpus.

Among all the baselines in Table 3.3, the `Ispell` spell checker (*SC*) outperforms language model-based approaches (*LM3* and *LM5*) in terms of F-score, but is inferior to the dictionary lookup method, and receives the lowest BLEU score of all methods over the SMS dataset. This suggests conventional off-the-shelf tools are often unsatisfactory in tweet normalisation.

Both web n-gram approaches are relatively ineffective at lexical normalisation. The primary reason for this can be attributed to the simplicity of the context modelling. Comparing the different-order language models, it is evident that longer n-grams (i.e., more highly-specified context information) support normalisation with higher precision. Nevertheless, lexical context in Twitter data is noisy: many OOV words are surrounded by Twitter user mentions, hashtags, URLs and other lexical variants, which are uncommon in other text genres. In the web n-gram approach, OOV words are mapped to the `<UNK>` flag in the `Web 1T` corpus construction process, leading to a loss of context information. Even the relaxed context constraints of the trigram method suffer from data sparseness, as indicated by the low recall. In fact, due to the temporal mismatch between the web n-gram corpus (harvested in 2006) and the Twitter data (harvested in late 2010), lexical variant contexts are often missing in the web n-gram data, limiting the performance of the web n-gram model for normalisation. Without the candidate filtering based on confusion sets, we observed that the web n-gram approach generated fluent-sounding normalisation candidates (e.g., *back, over, in, soon, home* and *events*) for *tomoroe* in *coming tomoroe* ("com-

Dataset	Evaluation	*SC*	*LM3*	*LM5*	*DL*	*NC*	*MT*	*WS*	*CS*	*WC*	*DWC*
SMS	P	0.209	0.116	0.556	**0.927**	0.465	—	0.521	0.116	0.532	0.756
	R	0.134	0.064	0.017	0.597	0.464	—	0.520	0.116	0.531	**0.754**
	F	0.163	0.082	0.033	0.726	0.464	—	0.520	0.116	0.531	**0.755**
	BLEU	0.607	0.763	0.746	0.801	0.746	0.700	0.764	0.612	0.772	**0.876**
Twitter	P	0.277	0.110	0.324	**0.961**	0.452	—	0.551	0.194	0.571	0.753
	R	0.179	0.068	0.020	0.460	0.452	—	0.551	0.194	0.571	**0.753**
	F	0.217	0.083	0.037	0.622	0.452	—	0.551	0.194	0.571	**0.753**
	BLEU	0.788	0.779	0.766	0.861	0.857	0.728	0.878	0.797	0.884	**0.934**

Table 3.3: Candidate selection effectiveness over different datasets (*SC* = spell checker; *LM3* = 3-gram language model; *LM5* = 5-gram language model; *DL* = dictionary lookup; *NC* = SMS noisy channel model (Cook and Stevenson 2009); *MT* = SMT (Aw *et al.* 2006); *WS* = word similarity; *CS* = context support; *WC* = WS + DS; *DWC* = DL + WS + DS).

ing tomorrow") but which lack semantic felicity with the original OOV word. This demonstrates the importance of candidate filtering as proposed.

The dictionary lookup method (*DL*) unsurprisingly achieves the best precision, but the recall on Twitter is moderate. Twitter normalisation cannot be tackled with such a small-scale dictionary lookup approach, although it is an effective preprocessing strategy when combined with other wider-coverage normalisation approaches (i.e., *DWC*). Nevertheless, because of the very high precision of the dictionary lookup method, we reconsider such an approach, but on a much larger-scale, in Section 3.3.

The spell checking-based noisy channel method of Cook and Stevenson (2009) (*NC*) shares similar features with our word similarity method (*WS*). However, when word similarity and context support are combined (*WC*), our method outperforms *NC* by about 7% and 12% in F-score over the SMS and Twitter datasets, respectively. This can be explained as follows. First, *NC* is type-based, so all token instances of a given lexical variant will have the same normalisation. However, the same lexical variant can correspond to different IV words, depending on context, e.g., *hw* "how" in *so hw many time remaining so I can calculate it?* vs. *hw* "homework" in *I need to finish my hw first*. Our word similarity method does not make the assumption that each lexical variant has a unique standard form. Second, *NC* was developed specifically for SMS normalisation, based on observations about how lexical variants are typically formed in text messages, e.g., clipping is fairly frequent in SMS. In Twitter, word lengthening for emphasis, such as *moviiie* "movie", is also common, but this is not the case in SMS; *NC* therefore performs poorly on such lexical variants.

The SMT approach is relatively stable on the two datasets, but performs well below our method. This is due to the limitations of the training data: because we don't have sufficient annotated Twitter data to train the SMT method directly, we obtain the lexical variants and their standard forms from the SMS corpus, but the lexical variants in the SMS corpus are not sufficient to cover those in the Twitter data (which reconfirms the empirical finding in Section 3.2.1 that lexical variants in Twitter are more diverse than in SMS). Thus, novel lexical variants are not recognised and are therefore not normalised. This shows the shortcomings of supervised data-driven approaches that require annotated data to cover an extensive range of lexical

variants in Twitter.

Of the component methods proposed in this research, word similarity (WS) achieves higher precision and recall than context support (CS), signifying that many of the lexical variants emanate from morphophonemic variations. However, when combined with context support, the performance improves over word similarity at a level of statistical significance (based on randomised estimation, $p < 0.05$: Yeh (2000)), indicating the complementarity of the two methods, especially on Twitter data. The best F-score is achieved when combining dictionary lookup, word similarity and context support (DWC), in which lexical variants are first looked up in the slang dictionary, and only if no match is found do we apply our normalisation method.

As is common in research on text normalisation (Choudhury *et al.* 2007; Liu *et al.* 2011a), throughout this section we have assumed perfect detection of lexical variants. This is, of course, not practical for real-world applications, and in the following section we consider the task of identifying lexical variants.

3.2.6 Lexical Variant Detection

A real-world end-to-end normalisation solution must be able to identify which tokens are lexical variants and require normalisation. In this section, we explore a context fitness-based approach for lexical variant detection. The task is to determine whether a given OOV word in context is a lexical variant or not, relative to its confusion set. To the best of our knowledge, we are the first to target the task of lexical variant detection in the context of Twitter, although related work exists for text with lower relative occurrences of OOV words (Izumi *et al.* 2003; Sun *et al.* 2007). Due to the noisiness of the data, it is impractical to use full-blown syntactic or semantic features. The most direct source of evidence is IV words around an OOV word. Inspired by work on labelled sequential pattern extraction (Sun *et al.* 2007), we exploit dependency-based features generated in Section 3.2.3.

We first present the high-level procedures for lexical variant detection and then discuss each step in detail.

- Train a binary classifier based on synthetic exemplars

  - Exemplars are dependency tuples such as `(book,hotel,−2)`.
  - Positive exemplars are from clean tweet data.
  - Negative exemplars are synthesised by substituting a correct word by similar words in its confusion set.

- Make binary predictions as to whether an OOV is a lexical variant or an unknown OOV
  - Generate a confusion set for an OOV in a tweet
  - For each word in the confusion set, extract dependency tuples and get classification results from the trained classifier
  - If the positive classification number exceeds a pre-defined threshold, then the OOV is considered to be a lexical variant; otherwise, it is skipped as an unknown OOV

To judge context fitness, we first train a linear kernel SVM classifier (Fan *et al.* 2008) on clean Twitter data, i.e., the subset of Twitter messages without OOV words (discussed in Section 3.2.3). Each target word is represented by a vector with dimensions corresponding to the IV words within a context window of three words to either side of the target, together with their relative positions in the form of `(target word,context word,position)` tuples, and with the feature value for a particular dimension set to the score for the corresponding tuple in the dependency bank.[14] These vectors form the positive training exemplars. Negative exemplars are automatically constructed by replacing target words with highly-ranked candidates from their confusion set. For example, we extract a positive instance for the target word *book* with a dependency feature corresponding to the tuple `(book,hotel,−2)`. A highly-ranked confusion of *book* is *hook* (Foster and Andersen 2009). We therefore form a negative instance for *hook* with a feature for the tuple `(hook,hotel,−2)`. In training, it is possible for the exact same feature vector to occur as both positive and negative exemplars. To prevent the positive exemplars from becoming contaminated

[14]Note that a dependency tuple only captures the paired words and their relative positions. It doesn't mean these tuples are generated by a dependency parser.

through the automatic negative-instance generation, we remove all negative instances in such cases. The `(target word,context word,position)` features are sparse and sometimes lead to conservative results in lexical variant detection. That is, without valid features, the SVM classifier tends to label uncertain cases as correct (i.e., not requiring normalisation) rather than as lexical variants. This is arguably the right approach to normalisation, in choosing to under- rather than over-normalise in cases of uncertainty. This artificially-generated data is not perfect; however, this approach is appealing because the classifier does not require any manually-annotated data, as all training exemplars are constructed automatically.

To predict whether a given OOV word is a lexical variant, we form a feature vector as above for each of its confusion candidates. If the number of the OOV's candidates predicted to be positive by the model is greater than a threshold t_d, we consider the OOV to be a lexical variant; otherwise, the OOV is deemed not to be a lexical variant. We experiment with varying settings of $t_d \in \{1, 2, ..., 10\}$. Note that in an end-to-end normalisation system, for an OOV predicted to be a lexical variant, we would pass all its confusion candidates (not just those classified positively) to the candidate selection step; however, the focus of this section is only on the lexical variant detection task.

As the context for a target word often contains OOV words which don't occur in the dependency bank, we expand the dependency features to include context tokens up to a phonemic edit distance of 1 from context tokens in the dependency bank. In this way, dependency-based features tolerate the noisy context word, e.g., given a lexical variant *seee*, its confusion candidate "see" can form `(see,film,+2)` in *film to seeee*, but not `(see,flm,+2)`. If we tolerate the context word variations assuming *flm* is "film", `(see,flm,+2)` would be also counted as `(see,film,+2)`. However, expanded dependency features may also introduce noise, and we therefore introduce expanded dependency weights $w_d \in \{0.0, 0.5, 1.0\}$ to ameliorate the effects of noise: a weight of $w_d = 0.0$ means no expansion, while 1.0 means expanded dependencies are indistinguishable from non-expanded (strict match) dependencies. Additionally, 0.5 represents a discounted weight for the counts of dependency tuples with noisy context words, in which the number of tuples are not fully counted, but are given a 0.5 weight penalty compared to the counts of standard dependency tuples.

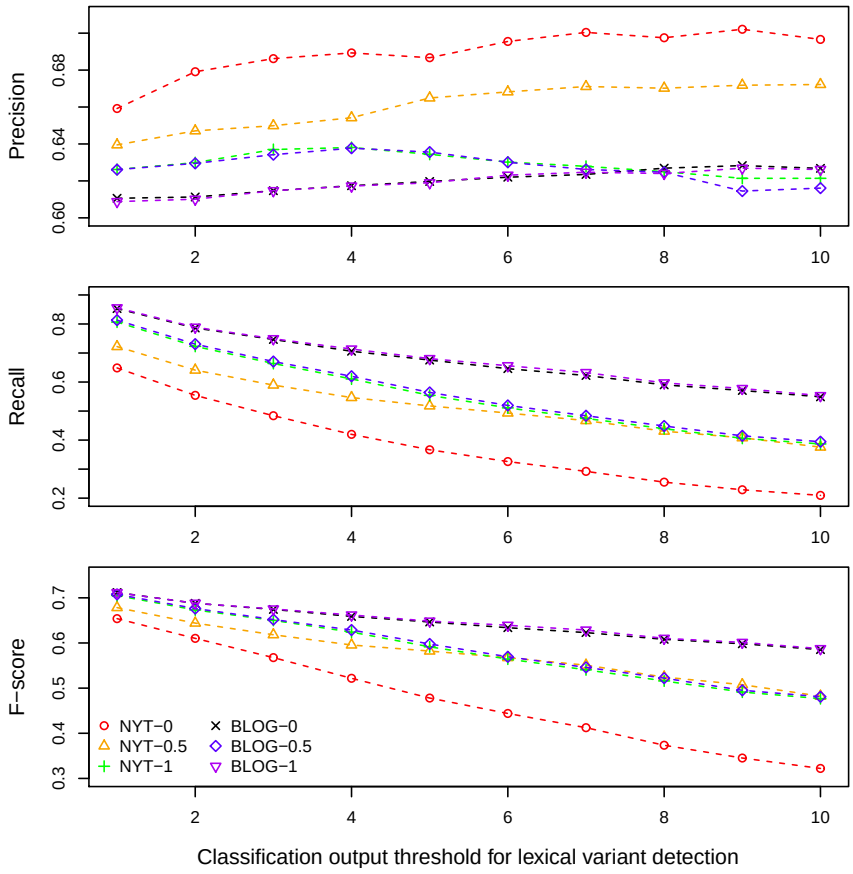


Figure 3.2: Lexical variant detection precision, recall and F-score.

We test the impact of the w_d and t_d values on lexical variant detection effectiveness for Twitter messages, based on dependencies from either the `NYT`, or the Spinn3r `BLOG` corpus (Burton *et al.* 2009), a large corpus of blogs which we also processed and parsed like Twitter data. The results for precision, recall and F-score are presented in Figure 3.2.

Some preliminary conclusions can be drawn from the graph. First, higher detection threshold values (t_d) give better precision but lower recall. Generally, as t_d is raised from 1 to 10, the precision improves slightly but recall drops dramatically, with

the net effect that the F-score decreases monotonically.[15] Thus, a smaller threshold (i.e., $t_d = 1$) is preferred. Second, there are differences between the two corpora, with dependencies from the BLOG producing slightly lower precision but higher recall, compared with the NYT. The lower precision for the BLOG appears to be due to the text not being as clean as NYT, introducing parser errors. Nevertheless, the difference between the two corpora with the best F-score is slight (when $t_d = 1$ and $w_d = 0.5$ on BLOG).

The lexical variant proportion among all OOV words in the Twitter dataset is 55%. Overall, the best F-score is 71.2%, with a precision of 61.1% and recall of 85.3%. Clearly there is significant room for improvement in these results. One quick solution is to find as many named entities as possible to filter out the non-lexical variant OOV words. Owning to the extensive editing, sheer volume of data and up-to-date content, we chose Wikipedia article titles as a source of standard OOV words which contains many named entities. However, the results in preliminary experiments with this data source did not lead to any improvement. By analysing the results, we found that terms from Wikipedia article titles are inappropriate for our task because they include many lexical variants such as *u* and *hw*, which decreased recall.

We further consider lexical variant detection based on the Internet slang dictionary in Section 3.2.4. In particular, if an OOV type has an entry in this dictionary, we consider all token instances of that type to be lexical variants; if a type is not in this dictionary, instances of that type are considered to be standard OOVs. This very simple method achieved precision, recall, and F-score of 95.2%, 45.3%, and 61.4%, respectively. Although the performance of this dictionary-based method is below that of our best-performing method in terms of F-score, we are encouraged by the very high precision of this method, particularly because of the previously-noted importance of not over-normalising.

[15] In most cases, one high-confidence positive candidate would be sufficient for detecting lexical variants, however, the number of positive candidates also matters in some cases. Given a tweet snippet *tak the lead*, *tak*'s confusion words may include "take", "takes", "taken" .... They may all form correct dependency tuples with "lead". As a result, a higher threshold number may give us higher detection accuracy than just one candidate fitting the context (which may be an inappropriate candidate that happens to form a correct dependency tuple with the tweet context).

Overall, the exploration of lexical variant detection suggests the task is challenging. The poor detection performance will negatively affect the overall normalisation performance. Nonetheless, the high precision of the dictionary-based method in both lexical variant detection and normalisation leads us to consider a type-based approach to normalisation.

3.2.7 Summary

In this section, our investigation started with a coarse-grained categorisation of lexical variants (i.e., OOV non-standard words) in Twitter, then we proposed a token-based normalisation approach using string and context information. A range of baselines and benchmarks were also compared such as methods based on machine translation and spell checking. These methods are not well suited to the normalisation task, either due to a lack of sufficient training data or because they are incapable of dealing with the wide variety of lexical variants in tweets. Overall the token-based approach outperformed these methods, however, it also has flaws in making an unrealistic assumption on the perfect detection of lexical variants. Further explorations on lexical variant detection suggest detection of lexical variants is a challenging task. As a result, the developed token-based method requires further improvement in tackling Twitter text normalisation.

The comparisons and analysis reveal the intrinsic difficulty of text normalisation. Nonetheless, we found a dictionary lookup approach to be very promising, owing to its speed and high end-to-end normalisation precision. The only downside is its less competitive recall in covering various types of lexical variants. As such, we are motivated to bridge this gap and develop a pure type-based normalisation approach in Section 3.3.

3.3 Type-based Lexical Normalisation

Given the appealing features of lexicon-based methods (e.g., high precision, integrated lexical variant detection and normalisation), we concentrate our explorations

on a type-based normalisation approach. In this section, we first discuss the feasibility of a pure lexicon-based approach for normalisation in Section 3.3.1. Then we introduce a two-step process for normalisation lexicon construction in Section 3.3.2: contextual similar (OOV, IV) pair generations and string similarity-based pair reranking are presented in Section 3.3.3 and Section 3.3.4, respectively. With the automatically-constructed lexicon, we then evaluate the type-based approaches and discuss experiment results in Section 3.3.5 and Section 3.3.6.

3.3.1 Motivation and Feasibility Analysis

Frequent (lexical variant, standard form) pairs such as (*u*, *you*) are typically included in existing dictionaries (e.g., Internet slang in Section 3.2.4), while less-frequent items such as (*g0tta*, *gotta*) are generally omitted. Because of the degree of lexical creativity and a large number of non-standard forms observed in Twitter, a wide-coverage normalisation dictionary would be expensive to construct manually. Based on the observation that lexical variants occur in similar contexts to their standard forms, it should be possible to automatically construct a normalisation dictionary with wider coverage than is currently available.

Dictionary lookup is a type-based approach to normalisation, i.e., every token instance of a given type will always be normalised in the same way. However, lexical variants can be ambiguous, e.g., *y* corresponds to "you" in *yeah, y r right! LOL* but "why" in *AM CONFUSED!!! y you did that?*. Nevertheless, the relative occurrence of ambiguous lexical variants is small (Liu *et al.* 2011a), and it has been observed that while shorter variants such as *y* are often ambiguous, longer variants tend to be unambiguous. For example *bthday* and *4eva* are unlikely to have standard forms other than "birthday" and "forever", respectively. Therefore, the normalisation lexicons we produce will only contain entries for OOVs with character length greater than a specified threshold, which are likely to have an unambiguous standard form.

Recently, Gouws *et al.* (2011a) produced a small normalisation lexicon based on distributional similarity and string similarity. Our method adopts a similar strategy using distributional/string similarity, but instead of constructing a small lexicon

for preprocessing, we build a much wider-coverage normalisation dictionary and opt for a fully lexicon-based end-to-end normalisation approach. In contrast with the normalisation lexicons which focus on very frequent lexical variants, we focus on moderate-frequency lexical variants of a minimum character length, which tend to have unambiguous standard forms; our intention is to produce normalisation lexicons that are complementary to those currently available. Furthermore, we investigate the impact of a variety of contextual and string similarity measures on the quality of the resulting lexicons. In summary, our dictionary-based normalisation approach pursues a lightweight end-to-end method which performs both lexical variant detection and normalisation, and thus is suitable for practical online preprocessing, despite its simplicity.

3.3.2 Word Type Normalisation

Our method for constructing a normalisation dictionary is as follows:

Input: Tokenised English tweets

1. Extract (OOV, IV) pairs based on distributional similarity.

2. Re-rank the extracted pairs by string similarity.

Output: A list of (OOV, IV) pairs ordered by string similarity; select the top-n pairs for inclusion in the normalisation lexicon.

In Step 1, we leverage large volumes of Twitter data to identify the most distributionally-similar IV type for each OOV type. The result of this process is a set of (OOV, IV) pairs, ranked by distributional similarity. The extracted pairs will include desired pairs such as (*tmrw*, *tomorrow*), but will also contain false positives such as (*Tusday*, *Sunday*) — *Tusday* is a lexical variant, but its standard form is not "Sunday" — and (*YouTube*, *web*) — *YouTube* is an OOV named entity, not a lexical variant. Nevertheless, lexical variants are typically formed from their standard forms through regular processes (Thurlow 2003; Cook and Stevenson 2009;

Xue *et al.* 2011), e.g., the omission of characters. The lexical variants and corresponding normalisations should be morphophonemically similar. From this perspective, *Sunday* and *web* are not plausible standard forms for *Tusday* and *YouTube*, respectively.

In Step 2, we therefore capture this intuition in re-ranking the extracted pairs by string similarity. The top-n items in this re-ranked list then form the normalisation lexicon, which is based only on development data. Although computationally-expensive to build, this dictionary can be created offline. Once built, it then offers a very fast approach to normalisation. However, this approach is not suitable for normalising low-frequency lexical variants, nor is it suitable for shorter lexical variant types which are more likely to have an ambiguous standard form. We can only reliably compute the distributional similarity for types that are moderately frequent in a corpus. Nevertheless, many lexical variants are sufficiently frequent to be able to compute their distributional similarity, and can potentially make their way into our normalisation lexicon. Furthermore, as the previously proposed token-based approach also relied in part on a normalisation lexicon, we can easily integrate the automatically-constructed lexicon with previous approaches to form hybrid normalisation systems.

3.3.3 Contextually Similar Pair Generation

Our objective is to extract distributionally-similar (OOV, IV) pairs from a large-scale collection of tweets. Fundamentally, the surrounding words define the primary context, but there are different ways of representing context and different similarity measures we can use (Lee 1999; Weeds *et al.* 2004), which may influence the quality of generated normalisation pairs.

Intuitively, distributional similarity measures the context proximity of two words in a corpus, as follows:

1. represent a word's context by its surrounding words in a (large) feature vector. Each entry in the vector represents a particular word, usually in the form of a

word frequency.[16]

2. calculate the similarity between two context vectors based on some distance/similarity measure. For instance, *tmrw* and *tomorrow* in Example (3.2) share a number of context words in the vector, like *see*, *you* and *school*, which suggests they are distributionally-similar.

(3.2) I don't wanna go to school *tmrw*
No school *tomorrow* or Tuesday woot!!!
okay off to work now . paipai . see you guys *tmrw* (:
ah i can't wait to see you *tomorrow*

In representing the context, we experimentally explore the following factors: (1) context window size (from 1 to 3 tokens on both sides); (2) n-gram order of the context tokens (unigram, bigram, trigram); (3) whether context words are indexed for relative position or not; and (4) whether we use all context tokens, or only IV words. Because high-accuracy linguistic processing tools for Twitter are still under exploration (Liu *et al.* 2011b; Gimpel *et al.* 2011; Ritter *et al.* 2011; Foster *et al.* 2011), we do not consider richer representations of context, for example, incorporating information about named entities or syntax. We also experiment with a number of simple but widely-used geometric and information theoretic distance/similarity measures. In particular, we use Kullback–Leibler (KL) divergence, Jensen–Shannon (JS) divergence (Lin 1991), Euclidean distance and Cosine distance.

We use a corpus of 10 million English tweets for parameter tuning, and a larger corpus of tweets in the final candidate ranking. All tweets were collected from September 2010 to January 2011 via the Twitter Streaming API. From the raw data we extract English tweets using an improved language identification tool `langid-2011` (Lui and Baldwin 2011),[17] and then apply a simplified Twitter tokeniser (adapted from O'Connor *et al.* (2010)). We again use the `Aspell` dictionary to determine whether

[16] Additionally, one can further apply stemming and Pointwise Mutual Information (PMI) (Church and Hanks 1989) to weight words. We leave these options to future work.

[17] A much-updated version of the language identification method used to construct the lexical normalisation dataset, trained over a larger sample of datasets, with feature selection based on the notion of domain generalisation.

a word is IV, and only include in our normalisation dictionary OOV tokens with at least 64 occurrences in the corpus and character length ≥ 4, both of which were determined through empirical observations. For each OOV word type in the corpus, we select the most similar IV type to form (OOV, IV) pairs. To further narrow the search space, we only consider IV words which are morphophonemically similar to the OOV type, based on parameter tuning from Section 3.2.3 over the top-30% of most frequent IV words in the confusion set.

In order to evaluate the generated pairs, we randomly selected 1000 OOV words from the 10 million tweet corpus. We set up an annotation task on Amazon Mechanical Turk,[18] presenting five independent annotators with each word type (with no context) and asking for corrections where appropriate. For instance, given *tmrw*, the annotators would likely identify it as a lexical variant of "tomorrow". For correct OOV words like *WikiLeaks*, on the other hand, we would expect them to leave the word unchanged. If 3 or more of the 5 annotators make the same suggestion (in the form of either a canonical spelling or leaving the word unchanged), we include this in our gold standard for evaluation. In total, this resulted in 351 lexical variants and 282 correct OOV words, accounting for 63.3% of the 1000 OOV words.[19] These 633 OOV words and annotator supplied IV words were used as (OOV, IV) pairs for parameter tuning. The remainder of the 1000 OOV words were ignored on the grounds that there was not sufficient consensus amongst the annotators.[20]

Contextually-similar pair generation aims to include as many correct normalisation pairs as possible. We evaluate the quality of the normalisation pairs using Cumulative Gain (CG):

$$\text{CG} = \sum_{i=1}^{N'} rel'_i$$

[18] `https://www.mturk.com/mturk/welcome`

[19] Disagreements between annotators are primarily caused by highly ambiguous lexical variants whose normalisations are potentially unknown. For instance, annotators considered *djing* as "darling", "disk jockeying", "Bing", or a correct named entity.

[20] Note that the objective of this annotation task is to identify lexical variants that have agreed-upon standard forms irrespective of context, as a special case of the more general task of lexical normalisation (where context may or may not play a significant role in the determination of the normalisation).

Rank	Window	n-gram	Positional?	Lex. choice	Sim./Dist.	log(CG)
1	±3	2	Yes	All	KL	19.571
2	±3	2	No	All	KL	19.562
3	±2	2	Yes	All	KL	19.562
4	±3	2	Yes	IV	KL	19.561
5	±2	2	Yes	IV	JS	19.554

Table 3.4: The five best parameter combinations in the exhaustive search of parameter combinations.

Suppose there are N' lexical variant and correction pairs (OOV_i, IV_i), each of which is weighted by rel'_i, the frequency of OOV_i to indicate its relative importance, e.g., (*thinkin*, *thinking*) has a higher weight than (*g0tta*, *gotta*) because *thinkin* is more frequent than *g0tta* in our corpus. In this evaluation we don't consider the position of normalisation pairs, and nor do we penalise incorrect pairs.[21] Instead, we push distinguishing between lexical variant and correct OOV pairs into the downstream re-ranking step in which we incorporate string similarity information.

Given the development data and CG, we run an exhaustive search of parameter combinations over our development corpus. The five best parameter combinations are shown in Table 3.4. We notice the CG is similar for the top combinations. As a context window size of 3 incurs a heavy processing and memory overhead over a size of 2, we use the 3rd-best parameter combination for subsequent experiments, namely: context window of ±2 tokens, token bigrams, positional index, and KL divergence as our distance measure.

To better understand the sensitivity of the method to each parameter, we perform a post-hoc parameter analysis relative to a default setting (as underlined in Table 3.5), altering one parameter at a time. The results in Table 3.5 show that bigrams outperform other n-gram orders by a large margin (note that the evaluation is based on a log scale), and information-theoretic measures are superior to the geo-

[21]Because we adopt the frequency of OOV as the weight, CGis therefore equivalent to recall.

Window size	n-gram	Positional?	Lexical choice	Similarity/Distance
±1 19.325	1 <u>19.328</u>	Yes **<u>19.328</u>**	IVs **19.335**	KL **<u>19.328</u>**
±2 19.327	2 **19.571**	No 19.263	All <u>19.328</u>	Euclidean 19.227
±3 **<u>19.328</u>**	3 19.324			JS 19.311
				Cosine 19.170

Table 3.5: Parameter sensitivity analysis measured as log(CG) for correctly-generated pairs. We tune one parameter at a time, using the default (<u>underlined</u>) setting for other parameters; the non-exhaustive best-performing setting in each case is indicated in **bold**.

metric measures. Furthermore, it also indicates that using positional indexing better captures context. However, there is little to distinguish context modelling with just IV words or all tokens. Similarly, the context window size has relatively little impact on the overall performance.

3.3.4 Pair Re-ranking by String Similarity

Once the contextually-similar (OOV, IV) pairs are generated using the selected parameters in Section 3.3.3, we further re-rank this set of pairs in an attempt to boost morphophonemically-similar pairs like (*bananaz*, *bananas*), and penalise noisy pairs like (*paninis*, *beans*).

Instead of using the small 10 million tweet corpus, from this step onwards, we use a larger corpus of 80 million English tweets (collected over the same period as the development corpus) to develop a larger-scale normalisation dictionary. This is because once pairs are generated, re-ranking based on string comparison is much faster. We only include in the dictionary OOV words with a token frequency > 15 to include more OOV types than in Section 3.3.3, and again apply a minimum length cutoff of 4 characters.

Given the generated pairs, we first consider three baselines: no re-ranking (i.e., the final ranking is that of the contextual similarity scores), and re-rankings of the

pairs based on the frequencies of the OOVs in the Twitter corpus, and the IV unigram frequencies in the `Web 1T` corpus (in Section 3.2.4) to get less-noisy frequency estimates. We also compared a variety of re-rankings based on a number of string similarity measures: standard edit distance; edit distance over Double Metaphone codes (phonetic edit distance: (Philips 2000)); longest common subsequence ratio over the consonant edit distance of the paired words (hereafter, denoted as consonant edit distance: (Contractor *et al.* 2010)); a string subsequence kernel (Lodhi *et al.* 2002), which measures common character subsequences of length n between (OOV, IV) pairs. Because it is computationally expensive to calculate similarity for larger n, we choose n=2, following Gouws *et al.* (2011a).

To measure how well our re-ranking method promotes correct pairs and demotes incorrect pairs (including both OOV words that should not be normalised, e.g., (*YouTube*, *web*), and incorrect normalisations for lexical variants, e.g., (*bcuz*, *cause*)), we modify our evaluation metric from Section 3.3.3 to evaluate the *ranking* at different points, using Discounted Cumulative Gain (DCG@N: Järvelin and Kekäläinen (2002)):

$$\text{DCG@}N = rel_1 + \sum_{i=2}^{N} \frac{rel_i}{\log_2(i)}$$

where rel_i again represents the frequency of the OOV, but it can be a gain (a positive number) or loss (a negative number), depending on whether the ith pair is correct or incorrect. Because we also expect correct pairs to be ranked higher than incorrect pairs, DCG@N takes both factors into account.

In Figure 3.3, we present the DCG@N results for each of our ranking methods at different rank cut-offs. Ranking by OOV frequency is motivated by the assumption that lexical variants are frequently used by social media users. This is confirmed by our findings that lexical pairs like (*goin*, *going*) and (*nite*, *night*) are at the top of the ranking. However, many proper nouns and named entities are also used frequently and ranked at the top, mixed with lexical variants like (*Facebook*, *speech*) and (*YouTube*, *web*). In ranking by IV word frequency, we assume the lexical variants are usually derived from frequently-used IV equivalents, e.g., (*abou*, *about*). However,

many less-frequent lexical variant types have high-frequency (IV) normalisations. For instance, the highest-frequency IV word *the* has more than 40 OOV lexical variants, such as *tthe* and *thhe*. These less-frequent types occupy the top positions, reducing the DCG@N. Compared with these two baselines, ranking by default contextual similarity scores delivers promising results. It successfully ranks many more intuitive normalisation pairs at the top, such as (*2day*, *today*) and (*wknd*, *weekend*), but also ranks some incorrect pairs highly, such as (*needa*, *gotta*).

The string similarity-based methods perform better than our baselines in general. Through manual analysis, we found that standard edit distance ranking is fairly accurate for lexical variants with low edit distance to their standard forms, e.g., (*thinkin*, *thinking*). Because this method is based solely on the number of character edits, it fails to identify heavily-altered variants like (*tmrw*, *tomorrow*). Consonant edit distance favours pairs with longer common subsequences, and therefore places many longer words at the top of the ranking. Edit distance over Double Metaphone codes performs particularly well for lexical variants that include character repetitions — commonly used for emphasis on Twitter — because such repetitions do not typically alter the phonetic codes. Compared with the other methods, the string subsequence kernel delivers encouraging results. As N (the lexicon size cut-off) increases, the performance drops more slowly than the other methods. Although this method fails to rank heavily-altered variants such as (*4get*, *forget*) highly, it typically works well for longer words. Given that we focus on longer OOVs (specifically those longer than 4 characters), this ultimately isn't a great handicap.

3.3.5 Intrinsic Evaluation of Type-based Normalisation

Given the re-ranked pairs from Section 3.3.4, we then apply them to a token-level normalisation task using the derived type-based lexicon, once again using the normalisation dataset from Section 3.2.2.

We use the same standard evaluation metrics of precision (P), recall (R) and F-score (F) as detailed in Section 3.2.2. In addition, we also consider the false alarm rate (FA) and word error rate (WER), as shown below. FA measures the negative effects

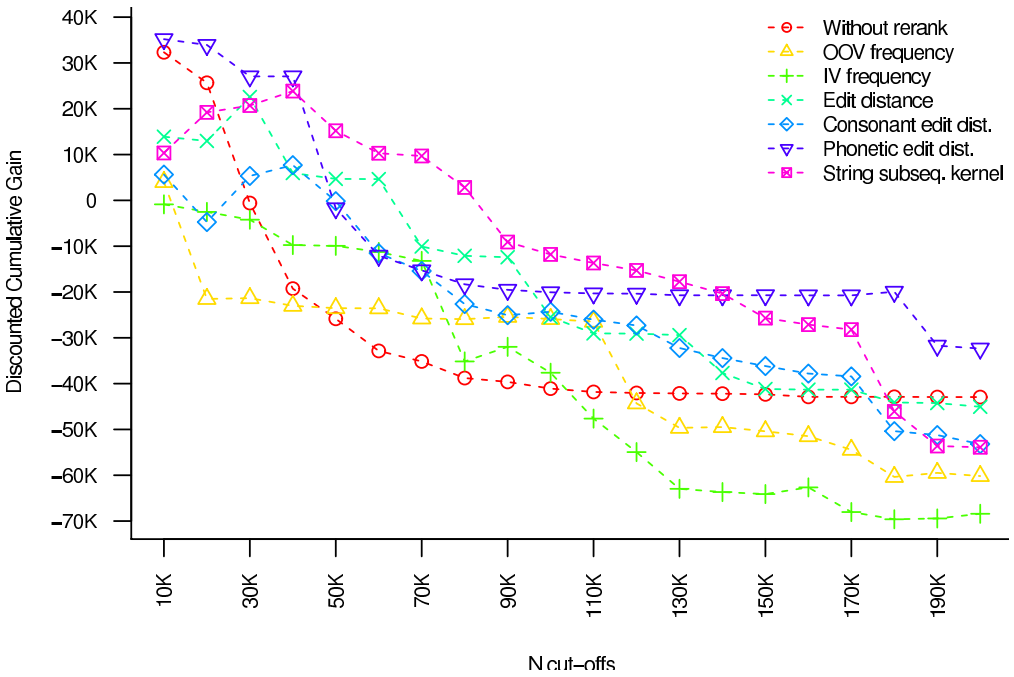


Figure 3.3: Re-ranking based on different string similarity methods.

of applying normalisation: a good approach to normalisation should not (incorrectly) normalise tokens that are already in their standard forms and do not require normalisation.[22] WER, like F-score, shows the overall benefits of normalisation, but unlike F-score, measures how many token-level edits are required for the output to be the same as the ground truth data. In general, dictionaries with a high F-score/low WER and low FA are preferable.

$$FA = \frac{\#\text{ incorrectly normalised tokens}}{\#\text{ normalised tokens}}$$

$$WER = \frac{\#\text{ token edits needed after normalisation}}{\#\text{ all tokens}}$$

[22] FA + P $\leq$ 1 because some lexical variants might be incorrectly normalised.

We select the three best re-ranking methods, and best cut-off N for each method, based on the highest DCG@N value for a given method over the development data, as presented in Figure 3.3.[23] Namely, they are string subsequence kernel (S-dict, N=40,000), Double Metaphone edit distance (DM-dict, N=10,000) and default contextual similarity without re-ranking (C-dict, N=10,000).[24]

We evaluate each of the learned dictionaries in Table 3.6. We also compare each dictionary with the performance of the manually-constructed Internet slang dictionary (HB-dict) used in Section 3.2.5, the small automatically-derived dictionary of Gouws *et al.* (2011a) (GHM-dict), and combinations of the different dictionaries. In addition, the contribution of these dictionaries in hybrid normalisation approaches is presented, in which we first normalise OOVs using a given dictionary (combined or otherwise), and then apply the normalisation method of Gouws *et al.* (2011a) based on consonant edit distance (GHM-norm), or the approach based on the summation of many unsupervised methods (HB-norm) in Section 3.2, to the remaining OOVs. Results are shown in Table 3.6, and discussed below.

Individual Dictionaries

Overall, the individual dictionaries derived by the re-ranking methods (DM-dict, S-dict) perform better than those based on contextual similarity (C-dict) in terms of precision and false alarm rate, indicating the importance of re-ranking. Even though C-dict delivers higher recall — indicating that many lexical variants are correctly normalised — this is offset by its high false alarm rate, which is particularly undesirable in normalisation. Because S-dict has better performance than DM-dict in terms of both F-score and WER, and a much lower false alarm rate than C-dict, subsequent results are presented using S-dict only.

Both HB-dict and GHM-dict achieve better than 90% precision with moderate

[23]The primary goal is to compare different re-ranking methods. Fine-grained cut-offs indeed generate lexicons with higher quality, but it is also computationally expensive to obtain the optimal lexicon.

[24]We also experimented with combining ranks using Mean Reciprocal Rank. However, the combined rank didn't improve performance on the development data. We plan to explore other ranking aggregation methods in future work.

Method	Precision	Recall	F-Score	False Alarm	Word Error Rate
C-dict	0.474	0.218	0.299	0.298	0.103
DM-dict	0.727	0.106	0.185	0.145	0.102
S-dict	0.700	0.179	0.285	0.162	0.097
HB-dict	0.915	0.435	0.590	0.048	0.066
GHM-dict	**0.982**	0.319	0.482	**0.000**	0.076
HB-dict+S-dict	0.840	0.601	0.701	0.090	0.052
GHM-dict+S-dict	0.863	0.498	0.632	0.072	0.061
HB-dict+GHM-dict	0.920	0.465	0.618	0.045	0.063
HB-dict+GHM-dict+S-dict	0.847	0.630	**0.723**	0.086	**0.049**
GHM-dict+GHM-norm	0.338	0.578	0.427	0.458	0.135
HB-dict+GHM-dict+S-dict+GHM-norm	0.406	0.715	0.518	0.468	0.124
HB-dict+HB-norm	0.515	0.771	0.618	0.332	0.081
HB-dict+GHM-dict+S-dict+HB-norm	0.527	0.789	0.632	0.332	0.079
HB-dict+GHM-dict+S-dict+HB-norm*	0.528	**0.791**	0.633	0.332	0.079

Table 3.6: Normalisation results using our derived dictionaries (contextual similarity (C-dict); Double Metaphone rendering (DM-dict); string subsequence kernel scores (S-dict)), the dictionary of Gouws *et al.* (2011a) (GHM-dict), the Internet slang dictionary (HB-dict) in Section 3.2.5, and combinations of these dictionaries. Furthermore, we combine the dictionaries with the normalisation method of Gouws *et al.* (2011a) (GHM-norm) and the combined unsupervised approach in (HB-norm) Section 3.2.3. In addition, we also compare context-sensitive normalisation on cleaned text after the dictionary lookup-based normalisation in the method suffixed with *.

recall. Compared to these methods, S-dict is not competitive in terms of either precision or recall. This result seems rather discouraging. However, considering that S-dict is an automatically-constructed dictionary targeting lexical variants of varying frequency, it is not surprising that the precision is worse than that of HB-dict — which is manually-constructed — and GHM-dict — which includes entries only for more-frequent OOVs for which distributional similarity is more accurate (e.g., *ppl* "people"). Additionally, the recall of S-dict is hampered by the restriction on lexical variant token length of 4 characters.

Combined Dictionaries

Next we turn to combining HB-dict, GHM-dict and S-dict. In combining the dictionaries, a given OOV word can be listed with different standard forms in different dictionaries. In such cases we use the following preferences to resolve conflicts: HB-dict > GHM-dict > S-dict. This order is motivated by the reliability of the dictionaries. The manually-constructed HB-dict is the most reliable dictionary, followed by GHM-dict which contains frequent normalisation pairs such as *ppl* "people". S-dict receives the lowest priority in the combination.

When we combine dictionaries in the second section of Table 3.6, we find that they contain complementary information: in each case the recall and F-score are higher for the combined dictionary than any of the individual dictionaries. The combination of HB-dict+GHM-dict produces only a small improvement in terms of F-score over HB-dict (the better-performing dictionary) suggesting that, as claimed, HB-dict and GHM-dict share many frequent normalisation pairs. HB-dict+S-dict and GHM-dict+S-dict, on the other hand, improve substantially over HB-dict and GHM-dict, respectively, indicating that S-dict contains markedly different entries to both HB-dict and GHM-dict. The best F-score and WER are obtained using the combination of all three dictionaries, HB-dict+GHM-dict+S-dict. Furthermore, the difference between the results using HB-dict+GHM-dict+S-dict and HB-dict+GHM-dict is statistically significant ($p < 0.01$), demonstrating the contribution of S-dict. We therefore use this best lexicon (i.e., HB-dict+GHM-dict+S-dict) for later intrinsic

and downstream evaluations (in Section 3.4).

Hybrid Approaches

So far, we have discussed a context-sensitive token-based normalisation in Section 3.2 and a type-based normalisation using a dictionary in this section. We therefore experiment with hybrid text normalisations. First, a lexicon is used to substitute lexical variants in the data, then the remaining lexical variants are normalised using two context-sensitive token-based normalisation systems: (1) Gouws *et al.* (2011a) (i.e., GHM-dict+GHM-norm); and (2) our proposed token-based hybrid approach (i.e., HB-dict+HB-norm).[25] Both methods have lower precision and higher false alarm rates than the dictionary-based approaches; this is largely caused by lexical variant detection errors.

Using all dictionaries in combination with these methods — HB-dict+GHM-dict+S-dict+GHM-norm and HB-dict+GHM-dict+S-dict+HB-norm — gives some improvement, but the false alarm rates remain high. In contrast, a larger dictionary-based method helps in improving the F-score and reducing the WER.

Impact of Context

As mentioned in Section 3.2.5, the disappointing performance of context features is partially attributable to noisy contexts, as neighbouring lexical variants mutually reduce the usable context of each other. To counter this effect, we apply context-sensitive token-based normalisation on the basis of the already partially normalised text (through our best dictionary) and compare its performance with token-based normalisation using the original unnormalised text, as shown in the last two rows of Table 3.6. This quantifies the relative impact of dictionary-based pre-normalisation on context-sensitive normalisation. An alternative way to examine the influence of context is by applying sequential labelling-based methods which capture the mutual influence of normalisations as in Section 2.2.3 on Page 39.

[25]We use the default settings in Gouws *et al.*'s (2011a) tool and the best context-based detection results in Section 3.2.6.

Data label	Messages
Noisy input message	*@USER damn that sucks im sorryy) :*
Normalisation on original message	*@USER damn that sucks him sorry) :*
Normalisation on cleaned message	*@USER damn that sucks i'm sorry) :*
Correct normalised message (oracle)	*@USER damn that sucks i'm sorry) :*

Table 3.7: An example where cleaned text helps context-sensitive normalisation.

The results indicate that partial pre-normalisation has only a very slight effect. Analysis of the two methods led to the finding that only 45 tokens were altered by the context-sensitive normalisation. That is, most lexical variants are already normalised by the lexicon in pre-normalisation, and it is not surprising that the context-sensitive lexical normalisation step had little impact.

We further analysed the 45 instances which the context-sensitive normalisation modified, and found that cleaned text does indeed help in context-sensitive normalisation, as shown in Table 3.7. When presented with the noisy context *sorryy*, the lexical variant *im* is incorrectly normalised to *him*, however, when the context is cleaned — i.e., *sorryy* is restored to *sorry* — *im* is correctly normalised to "i'm", as both the language model-based and dependency-based context feature strongly support the usage of "i'm sorry".

Despite the limitations of a pure dictionary-based approach to normalisation — discussed in Section 3.3.1 — the best practical approach to normalisation is to use a lexicon, combining hand-built and automatically-learned normalisation dictionaries.

3.3.6 Error Analysis and Discussion

We manually analyse the errors in the combined dictionary (HB-dict+GHM-dict+S-dict) and give examples of each error type. As shown in Table 3.8, many types of word errors are caused by slight morphological variations. For countable nouns (a), whether to use a plural or a singular form requires contextual inference, although its correct form is sometimes difficult to determine from noisy tweet text.

Error type	OOV	Standard form	
		Dict.	Gold
(a) plurals	*playe*	*players*	*player*
(b) negation	*unliked*	*liked*	*disliked*
(c) possessives	*anyones*	*anyone*	*anyone's*
(d) correct OOVs	*iphone*	*phone*	*iphone*
(e) annotation errors	*durin*	*during*	*durin*
(f) ambiguity	*siging*	*signing*	*singing*

Table 3.8: Error types in the combined dictionary (HB-dict+GHM-dict+S-dict).

Negation errors (b) and possessive errors (c) appear to be caused by the low proficiency of English by users whose native language is not English. There also exist correct OOVs that are over-normalised (d), which is largely due to the small and out-dated IV lexicon. In addition, we also notice some missing annotations where lexical variants are skipped by human annotators but are captured by our method (e). Ambiguity (f) exists in longer OOVs, however, these cases do not appear to have a strong negative impact on the normalisation performance. An example of a remaining miscellaneous error is *bdayy*, which is mis-normalised as "day", instead of "birthday".

To further study the influence of OOV word length relative to the normalisation performance, we conduct a fine-grained analysis of the performance of the derived dictionary (S-dict) in Table 3.9, broken down across different OOV word lengths. The results generally support our hypothesis that our method works better for longer OOV words. The derived dictionary is much more reliable for longer tokens (length 5, 6, and 7 characters) in terms of precision and the false alarm rate. Although the recall is relatively low, there is still room for further improvement, either by mining more normalisation pairs from larger collections of microblog data or by exploiting context transitivity discussed in Section 2.2.4.

In addition, we further get the statistics from our dataset.[26] We analysed lexical

[26]This dataset merges revisions from (Yang and Eisenstein 2013) described in Section 3.6.1.

Cut-off (N)	#Variants	Precision	Recall ($\geq N$)	Recall (all)	False Alarm
$\geq$4	556	0.700	0.381	0.179	0.162
$\geq$5	382	0.814	0.471	0.152	0.122
$\geq$6	254	0.804	0.484	0.104	0.131
$\geq$7	138	0.793	0.471	0.055	0.122

Table 3.9: S-dict normalisation results broken down according to OOV token length. Recall is presented both over the subset of instances of length $\geq N$ in the data ("Recall ($\geq N$)"), and over the entirety of the dataset ("Recall (all)"); "#Variants" is the number of token instances of the indicated length in the test dataset.

variants of all lengths to calculate: (a) the proportion of word types of a given length which are ambiguous in the dataset; and (b) the proportion of tokens of a given length which have ambiguous types. We found that for types of 1, 2 and 3, the proportion of ambiguous types was 29.4%, 6.3% and 0.9%, respectively; in terms of word tokens, the respective proportions length are 28.9%, 5.0% and 2.1%, respectively. All lexical variants more than 3 characters in length were unambiguous.

3.4 Extrinsic Evaluation of Lexical Normalisation

Having proposed a number of approaches to lexical normalisation and evaluating those methods directly, we now evaluate the impact of normalisation in an applied setting. When existing NLP tools trained on more conventional text are applied to social media, their performance is hampered in part due to the presence of lexical variants as discussed in Section 2.2.1 on Page 29. We therefore hypothesise that the performance of such tools might improve if lexical normalisation is applied after tokenisation, and before subsequent processing.

In this section we test the above hypothesis on a Twitter part-of-speech (POS) tagging task. Many NLP tasks and downstream applications can be equipped with normalisation modules. We choose POS tagging for the following reasons: (1) the

impact of lexical normalisation is readily-observed, as it is easy to compare the POS tags for the original and normalised texts; (2) off-the-shelf part-of-speech taggers are available for both more-conventional text (Toutanova *et al.* 2003) and social media (Gimpel *et al.* 2011); and (3) a human-annotated Twitter POS tagging dataset `TW-POS` is publicly available (Gimpel *et al.* 2011).[27]

`TW-POS` consists of 1827 tokenised and annotated messages from Twitter. 500 tweets — referred to as the test set — are held out for test purposes, with the rest of the data being used for training and development, as described in Gimpel *et al.* (2011). For each message in the test set, we apply the best-performing dictionary-based normalisation method from Section 3.3.5, namely HB-dict+GHM-dict+S-dict.

When substituting words, we also consider the capitalisation information of original tokens, as this information is known to be important for POS tagging (Gimpel *et al.* 2011) and named entity recognition (Ritter *et al.* 2011). Specifically, the case of the first and last characters of the normalised word form are set to the case of the first and last characters of the original token, respectively. All the other characters of the normalised form are set to the case of the middle character of the original token. For example, *Todei*, *WKEND* and *tmrw* are normalised as "Today", "WEEKEND" and "tomorrow", respectively.

We compare the performance of the Twitter-specific POS tagger ("$\text{POS}_{\text{Twitter}}$": Gimpel *et al.* (2011)) to that of a standard off-the-shelf tool, the Stanford POS tagger ("$\text{POS}_{\text{Stanford}}$": Toutanova *et al.* (2003)). However, these taggers use different tagsets: $\text{POS}_{\text{Twitter}}$ uses a much more coarse-grained tagset than the Penn Treebank POS tagset that is used by $\text{POS}_{\text{Stanford}}$.[28] We are primarily interested in the performance comparison of a conventional off-the-shelf tool on raw and cleaned tweets, and therefore do not re-train $\text{POS}_{\text{Stanford}}$ on the POS-annotated tweets.

To bridge the tagset difference, we manually devised a lossy mapping from the fine-grained $\text{POS}_{\text{Stanford}}$ tagset to that of $\text{POS}_{\text{Twitter}}$. In this mapping, finer-grained tags unique to $\text{POS}_{\text{Stanford}}$ (e.g., *VBP* and *VBN*) are mapped to coarser-grained $\text{POS}_{\text{Twitter}}$

[27] `http://ark-tweet-nlp.googlecode.com/files/twpos-data-v0.2.tar.gz`

[28] At the time of experiments, a Penn Treebank POS tagset was not available in $\text{POS}_{\text{Twitter}}$. Recently, $\text{POS}_{\text{Twitter}}$ was updated to support the Penn Treebank POS tagset `http://www.ark.cs.cmu.edu/TweetNLP/model.ritter_ptb_alldata_fixed.20130723`

Tagger	Text	% accuracy	# correct tags
$POS_{Stanford}$	original	75.6	5414
$POS_{Stanford}$	normalised	77.2	5527
$POS_{Twitter}$	original	95.2	6819
$POS_{Twitter}$	normalised	94.8	6790
$POS_{MostFreq}$	original	79.5	5697
$POS_{MostFreq}$	normalised	79.9	5723

Table 3.10: Comparison of accuracy of $POS_{Stanford}$ (a general-purpose POS tagger), $POS_{MostFreq}$ (a most frequent tag baseline) and $POS_{Twitter}$ (a Twitter POS tagger) applied to the original and normalised tweets in the test set. The total number of correct tags is also shown.

tags (e.g., *V*).[29]

We apply $POS_{Twitter}$ and $POS_{Stanford}$ to the test set, both with and without first applying normalisation. Additionally, we experimented with a most frequent tag baseline (i.e., $POS_{MostFreq}$) as follows: We tag words in test data using the most frequent tags in the training data, and if a word is not seen in the training data, it is tagged as a noun. We use accuracy to measure performance. The Twitter-specific POS tags are copied from the gold-standard in the calculation (as there is no way of reliably mapping onto them from the Penn POS tagset).

Results are shown in Table 3.10. First, we compare the performance of $POS_{Stanford}$ on the original tweets to its performance on the normalised tweets. The accuracy on normalised text is 1.6 percentage points higher than that on the original text. In total, 113 more tokens are correctly tagged when lexical normalisation is used. We observe that most of this improvement is for nouns and verbs. Although the improvement in performance is small, it is statistically significant ($p < 0.01$). Furthermore, only

[29]The test set provides tokenised tweets, but contains some tokenisation errors, e.g., *"Success* is tokenised as a single token, instead of as the pair of tokens *"* and *Sucess*. In the small number of such cases we manually correct the tokens output from the POS tagger to be consistent with the test set tokenisation.

264 tokens in the test set are normalised (i.e., the remaining tokens are unchanged through normalisation). It could be the case that more normalisation would lead to a greater improvement in POS tagging performance for noisier text containing more lexical variants.

We further consider the impact of normalisation on $\text{POS}_{\text{Twitter}}$, the Twitter-specific tagger. In this case the performance on pre-normalised tweets drops slightly over that for the original messages, indicating that normalising the input hurts the performance of $\text{POS}_{\text{Twitter}}$. This is somewhat expected because some features used by $\text{POS}_{\text{Twitter}}$ are derived from noisy tokens: when the input is normalised, some of these features are not present, e.g., features capturing co-occurrence with lexical variants.

As for $\text{POS}_{\text{MostFreq}}$, applying normalisation also improves accuracy, although only 26 additional tokens are recognised and matched in the most frequent tag dictionary after the normalisation. $\text{POS}_{\text{MostFreq}}$ outperforms $\text{POS}_{\text{Stanford}}$ by 3%, which confirms the noise challenge in Twitter data. Features defined on conventional text may degrade the performance of off-the-shelf tools on noisy tweets, e.g., $\text{POS}_{\text{Stanford}}$ misses many proper nouns and incorrectly tags many other words as proper nouns, due to unreliable capitalisations in tweets. Nonetheless, we believe that by dropping such features, $\text{POS}_{\text{Stanford}}$ would outperform $\text{POS}_{\text{MostFreq}}$.[30]

Interestingly, we find $\text{POS}_{\text{Twitter}}$ outperforms $\text{POS}_{\text{Stanford}}$ and $\text{POS}_{\text{MostFreq}}$ by a large margin. This finding makes sense, because lexical normalisation aims to reduce the variance of words to make the data more accessible to existing NLP tools. However, lexical variance is only one source of noise in tweets. As discussed in Section 2.2.1 on Page 29, other sources of noise such as ungrammatical sentence structure and Twitter entities (e.g., #hashtags and @USER) contribute to the degradation of NLP tools as well. In contrast, the Twitter POS tagger is trained with supervision specific to Twitter and carefully engineered features, and therefore has a distinct advantage.

In summary, these preliminary comparisons show the influence of normalisation on the task of POS tagging for Twitter. In terms of cost, applying normalisation to a

[30] An improved caseless model for the English POS tagger was also released recently: `http://nlp.stanford.edu/software/stanford-corenlp-caseless-2013-11-12-models.jar` (Retrieved 01/14).

conventional off-the-shelf tool (e.g., $\text{POS}_{\text{Stanford}}$) or a most frequent tag baseline (e.g., $\text{POS}_{\text{MostFreq}}$) is the cheapest option, and would obviate the need for the development of a Twitter-specific tool such as $\text{POS}_{\text{Twitter}}$. Not surprisingly, building a tool specific to the target domain yields the best performance. However, this comes at the substantial overhead of developing a specific tagset, manually annotating training data, and developing the tagger itself. Furthermore, developing domain-specific tools is generally task dependent, i.e., training a POS tagger doesn't make domain-specific NER work. In contrast, text normalisation over the target domain is generally universal, reducing OOV rates for all downstream NLP tasks and applications.

3.5 Non-English Text Normalisation

So far our discussion has been exclusively based on English tweets. Notably, non-English tweets such as Spanish (Alegria *et al.* 2013) and Chinese (Wang and Ng 2013) also suffer from the negative impact of lexical variants. The formation of lexical variants is different across languages, and therefore text normalisation is also language dependent. For instance, phonetic approximations are more popular in Chinese than in English (Xia *et al.* 2006; Wang and Ng 2013). Furthermore, methods like character edit distance in English are not applicable to Chinese words. Nevertheless, it is plausible to adapt the same type-based normalisation approach for languages that are similar to English.

In this section, we adapt our approach to Spanish text normalisation in the `TWEET-NORM` shared task.[31] First a brief comparison between English and Spanish is presented in Section 3.5.1. Then we summarise the collected resources and methods for Spanish normalisation in Section 3.5.2. Finally, we analyse the experimental results and present our discussion in Section 3.5.3.

[31] `http://komunitatea.elhuyar.org/tweet-norm/`

3.5.1 A Comparative Study on Spanish Text Normalisation

In this section we consider the plausibility of adapting the lexicon-based method in Section 3.3 from English to Spanish, and identify the following key factors:

Orthography: If we consider diacriticised letters as single characters, Spanish has more characters than English, and diacritics can lead to differences in meaning, e.g., *más* means "more", and *mas* means "but". The method in Section 3.3 uses edit distance to measure string similarity. We simply convert all characters to fused Unicode code points (treating *á* and *a* as different characters) and compute edit distance over these forms.

Word segmentation: Spanish and English words both largely use whitespace segmentation, so similar tokenisation strategies can be used.

Morphophonemics: Phonetic modelling of words is also available for Spanish using an off-the-shelf Double Metaphone implementation.[32]

Lexical resources: A lexicon and slang dictionary — key resources for the method in Section 3.3 — are available for Spanish.

The `TWEET-NORM` task setting is also similar to the setting in Section 3.1: transforming non-standard spellings of OOV words to standard IV words. However, there are some differences between the two task settings. First, instead of being restricted to one-to-one normalisation, `TWEET-NORM` allows one-to-many mappings such as *mencantaba* "me encantaba". Another important component of `TWEET-NORM` task is case restoration: e.g., *maria* as a name should be normalised to "Maria". Most previous English Twitter normalisation tasks have ignored capitalisation.

Nonetheless, English and Spanish text share important features, and we hypothesise that adapting a lexicon-based English normalisation system to Spanish is feasible.

[32] `https://github.com/amsqr/Spanish-Metaphone`

3.5.2 Adapted Normalisation Approach

The system consists of two steps: (1) down-case all OOVs and normalise them based on a normalisation lexicon which combines entries from existing lexicons and entries automatically learnt from a Twitter corpus; and (2) restore case for normalised words.

Resources

`TWEET-NORM` offers 500 and 564 Spanish tweets for development and test data. In each tweet, OOV words are marked and categorised into proper nouns (including neologism and foreign words) or non-standard words.[33] Each non-standard OOV is accompanied with a correct normalised IV term.

Our normalisation transforms lexical variants (i.e., OOV non-standard words) to IV words, and thus a Spanish dictionary is required to determine what is OOV. To this end, we use the `Freeling 3.0` (Padró and Stanilovsky 2012) Spanish dictionary, which contains 669K words.

We collected 146 Spanish Internet slang expressions and cell phone abbreviations from the web (`Slang Lexicon`).[34] We further extracted normalisation pairs from the development data (`Dev Lexicon`) in `TWEET-NORM`.

Through analysing `Dev Lexicon`, we noticed that many person names are not correctly capitalised. We formed `Name Lexicon` from a list of 277 common Spanish names.[35] This lexicon maps lowercase person names to their correctly capitalised forms.

Corpus-derived Lexicon

The small, manually-crafted normalisation lexicons have low coverage over lexical variants. To improve coverage, we automatically derive a much larger normalisation lexicon based on distributional similarity (`Dist Lexicon`) by adapting the method in

[33]The OOV words are identified relative to the dictionary of Real Academia Española.
[34]`http://goo.gl/wgCFSs` and `http://goo.gl/xsYkDe` (Retrieved 06/2013)
[35]`https://en.wikipedia.org/wiki/Spanish_naming_customs` (Retrieved 06/2013)

Section 3.3.

We collected 283 million Spanish tweets via the Twitter Streaming API from 21/09/2011–28/02/2012. Spanish tweets were identified using `langid-2012` (Lui and Baldwin 2012). The tweets were tokenised using the same English Twitter tokeniser (O'Connor *et al.* 2010) as in Section 3.3.3. Excessive repetitions of characters (i.e., ≥ 3) in words are shortened to one character to ensure different variations of the same pattern are merged. To improve coverage, we removed the restriction from the original work that only OOVs with ≥ 4 letters were considered as candidates for normalisation.

For a given OOV, we define its confusion set to be all IV words with edit distance ≤ 2 in terms of characters or ≤ 1 in terms of Double Metaphone code in line with settings in Section 3.2.3. We rank the items in the confusion set according to their distributional similarity to the OOV. Instead of experimenting with many configurations of distributional similarity for normalisation, we use the same optimised settings as the English data: context is represented by positionally-indexed bigrams using a window size of ± 2 tokens; and similarity is measured using KL divergence.[36] An entry in the normalisation dictionary then consists of the OOV and its top-ranked IV.

From the development data, we observe that in many cases when a correct normalisation is identified, there is a large difference in KL divergence between the first- and second-ranked IVs. Conversely, if the KL divergence of the first- and second-ranked normalisation candidates is similar, the normalisation is often less reliable. As shown in Table 3.11, *callendo* "cayendo" is a correctly-derived (OOV, IV) pair, but *guau* "y" is not.

Motivated by this observation, we filter the derived (OOV, IV) pairs by the KL divergence ratio of the first- and second-ranked IV words for the OOV. Setting a high threshold on this KL divergence ratio increases the reliability of the derived lexicon, but reduces its coverage. This ratio was tested for values from 1.0 to 3.0 with a step size of 0.1 over the development data and the `Slang Lexicon`. As shown in

[36] We used additive smoothing with $\alpha = 10^{-6}$ when calculating KL divergence.

Rank	*callendo*		*guau*	
1	*cayendo*	0.713	*y*	1.756
2	*saliendo*	3.896	*que*	1.873
3	*fallando*	4.303	*la*	2.488
4	*rallando*	6.761	*a*	2.649
5	*valiendo*	6.878	*no*	3.206

Table 3.11: The KL divergence for the top-five candidates for *callendo* and *guau*.

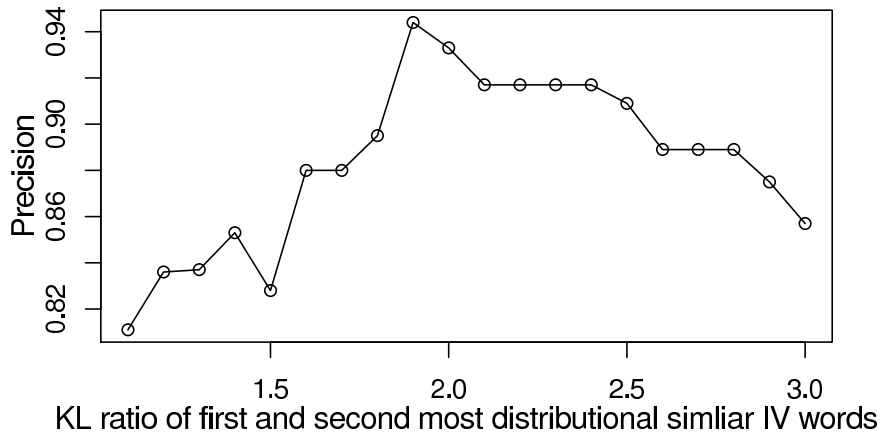


Figure 3.4: KL divergence ratio cut-off vs. precision of the derived normalisation lexicon on the development data and `Slang Lexicon`.

Figure 3.4, the best precision (94.0%) is achieved when the ratio is 1.9.[37] We directly use this setting to derive the final lexicon, instead of further re-ranking the (OOV,IV) pairs using string similarity.

Case Restoration

We set the case of each token that was normalised in the previous step (which is down-cased at the current stage) to its most-frequent casing in our corpus of Spanish tweets. We also trivially capitalise all normalised tokens occurring at the beginning

[37]Here precision is defined as $\frac{\#\text{correct normalisations}}{\#\text{normalisations}}$.

Lexicon	Accuracy
`Combined Lexicon`	0.52
− `Slang Lexicon`	0.51
− `Dev Lexicon`	0.46
− `Dist Lexicon`	0.42
− `Name Lexicon`	0.51
+ Edit distance	0.54
Baseline	0.20

Table 3.12: Accuracy of lexicon-based normalisation systems. "−" indicates the removal of a particular lexicon.

of a tweet, or following a period or question mark.

3.5.3 Results and Discussion

We evaluated the lexicons using classification accuracy, the official metric for this shared task, on the `TWEET-NORM` test data. This metric divides the number of correct proposals — OOVs correctly normalised or left unchanged — by the number of OOVs in the collection. This is termed "precision" by the task organisers, but a true measure of precision would be based on the number of OOVs that were actually normalised. We therefore use the term "accuracy" here.

We submitted two runs for the task. The first, `Combined Lexicon` (Table 3.12), uses only the combination of lexicons from Section 3.5.2, and achieves an accuracy of 0.52. The second run builds on `Combined Lexicon` but incorporates normalisation based on character edit distance for words with many repeated characters. We observed that such words are often lexical variants, and tend not to occur in the lexicons because of their relatively low frequency. For words with ≥ 3 repeated characters, we remove all but one of the repeated characters, and then select the most similar IV word according to character-based edit distance. The accuracy of this run is 0.54 (+ Edit distance, Table 3.12).

We further consider an ablative analysis of the component lexicons of `Combined Lexicon`. As shown in Table 3.12, when `Slang Lexicon` (− `Slang Lexicon`) or `Name Lexicon` (− `Name Lexicon`) are excluded, accuracy declines only slightly. Although this suggests that existing resources play only a minor role in the normalisation of Spanish tweets, this is likely due in part to the relatively small size of `Slang Lexicon`, which is much smaller than similar English resources that have been effectively exploited in normalisation — i.e., 145 Spanish entries versus 5K English entries used in Section 3.2.4. Furthermore, `Slang Lexicon` might have little impact due to differences between Spanish Twitter and SMS, the latter being the primary focus of `Slang Lexicon`.

On the other hand, normalisation lexicons derived from tweets — whether based on the development data (`Dev Lexicon`) or automatically learnt (`Dist Lexicon`) — substantially impact on accuracy (− `Dev Lexicon` and − `Dist Lexicon`). These findings for the automatically-derived `Dist Lexicon` are in line with previous findings for English Twitter normalisation in Section 3.3.5 that indicate that such lexicons can substantially improve recall with little impact on precision.

We considered an experiment in which we used `Combined Lexicon`, but ignored case in the evaluation; the accuracy was 0.56. This corresponds to the upper-bound on accuracy if our system performed case restoration perfectly, and suggests that improving the case restoration of our system would not lead to substantial gains in accuracy.

In the final row of Table 3.12, we show results for a baseline method which makes no attempt to normalise the input. All lexicon-based methods improve substantially over this baseline.

To further analyse our lexicon-based normalisation approach, we categorise the errors for both false positives (OOVs that were normalised, but incorrectly so) and false negatives (OOVs that were not normalised, but should have been). As shown in Table 3.13, 37% of false positives are incorrect lexical forms, e.g., *algerooo* is normalised to "algero" and not its correct form "alegra". Further examination shows that 23% of these cases are incorrectly normalised to "que", suggesting that distributional similarity alone is insufficient to capture normalisations for lexical variants.

Error type	Number	Percentage
Incorrect lexical form	22	37%
Not available	19	32%
Accent error	10	17%
Case error	5	8%
One to many	2	3%
Annotation error	1	2%

Table 3.13: Categorisation of false positives.

Surprisingly, we found some OOVs included in the test data, but excluded from the gold-standard annotations (due to tweet deletions), or present in the test data, but not found in the tweets, and excluded in the gold standard. These error types are denoted as "Not available" in Table 3.13, and account for the second largest source of false positives.

Incorrect accents and casing account for 17% and 8% of false positives, respectively. In both of these cases, contextual information, which is not incorporated in the proposed approach, could be helpful. Finally, we identified two one-to-many normalisations (which are outside the scope of our normalisation system), and one case we judged to be an annotation error.

We analysed a random sample of 20 of the 280 false negatives, and found irregular character repetitions and named entities to be the main sources of errors, e.g., *uajajajaa* "ja" and *Pedroo* "Pedro".[38] The lexicon-based approach could be improved, for example, by using additional regular expressions to capture repetitions of character sequences. Errors involving named entities reveal the limitations of using the `Freeling 3.0` Spanish dictionary as the IV lexicon, as it has limited coverage of named entities. A corpus-derived lexicon (e.g., from Wikipedia) could help improve the coverage.

In summary, we applied a type-based approach to normalise lexical variants in

[38] *Pedro* is not in our collected list of Spanish names.

Spanish tweets using various lexicons. Our findings are in line with English text normalisation results. The results suggest that the corpus-derived lexicon based on distributional similarity improves accuracy, but that this approach is limited in terms of flexibility (e.g., to capture accent variation) and lexicon coverage (e.g., of named entities). The best system on this shared task achieved 0.78 accuracy, which outperforms our system by a large margin (Porta and Sancho 2013). They applied finite state transducers to propose potential normalisations and select the most probable normalisation based on word trigrams. Additionally, they also used a plethora of lexical resources including the `DRAE` dictionary and frequent English words from the BNC corpus.[39] The language model is built on basis of a corpus compiled form web pages. The large gap between our adapted system and the best system suggests there is room for improvement. In the future, we plan to expand the IV lexicon, and incorporate contextual information to improve normalisation involving accents and casing.

3.6 Recent Progress on Text Normalisation

As discussed in Section 2.2.4, many methods emerged after we developed the type-based approach in Section 3.3. In this section, we summarise and compare recent methods for text normalisation. In particular, we survey recent related methods and normalisation results that have been reported in the literature, and benchmarked against our combined lexicon from Section 3.3 or their in-house normalisation systems.

3.6.1 Recent Normalisation Approaches

The recent approaches described in Section 2.2.4 primarily extend our work in the following ways: (1) the generation of normalisation candidates integrating various sources of information such as a spell checker and Internet slang; (2) they allow more flexible context-sensitive normalisation, and moreover the normalisations are jointly determined, instead of being selected independently; (3) they incorporate context transitivity in normalisation candidate generation, i.e., by using proxy context

[39] `http://www.rae.es/recursos/diccionarios/drae`

words, two contextually similar tokens can be associated even though they don't directly share common context; and (4) they source large-scale training data, e.g., using paraphrasing to generate parallel data for machine translation-based methods.[40]

Only a handful of recent work (Hassan and Menezes 2013; Yang and Eisenstein 2013) has been evaluated against our off-the-shelf dataset. Hassan and Menezes (2013) compared their approach with our automatically-constructed lexicon (without using the existing lexicons), and achieved much better results in terms of both precision and recall. However, the proposed approach is outperformed by the best lexicon combining both automatically-derived and existing lexicons. Yang and Eisenstein (2013) also reported results on the same dataset.[41] Relative to our best lexicon, the recall in their approach was improved by approximately 19% with a slight dip (2%) in precision. Recently, Chrupała (2014) achieved a lower word error rate by incorporating simple recurrent networks to model string transformation from unlabelled tweets. Nonetheless, the best lexicon developed in Section 3.3 is the fastest lightweight end-to-end normalisation solution, with reasonable precision. We believe these features are important and practical to processing large-scale social media data. In the next section, we report recent results of applying the best lexicon in NLP tasks and downstream applications. We also outline results using in-house normalisation modules.

3.6.2 Impact of Normalisation in Recent Research

Recently, our combined lexicon (i.e., HB-dict+GHM-dict+S-dict from Section 3.3) has been reported to achieve encouraging results in a range of downstream NLP tasks and applications. For instance, Xu *et al.* (2013) obtained a 15% increase in BLEU in a machine translation task using their independently collected tweet data. Hassan and Menezes (2013) showed the BLEU in an in-house machine translation system

[40]The biggest improvements are made through a combination of these approaches, but it is interesting to explore the independent contribution of each method.

[41]They also offered a slightly corrected version of the dataset described in Section 3.2.2, which has since been merged with the original dataset into an updated version of the dataset `http://www.csse.unimelb.edu.au/~tim/etc/lexnorm_v1.2.tgz`. Their results on the two versions of the data are indistinguishable.

is improved by 3.69% over a benchmark method, with an absolute improvement in BLEU of 0.74%. Note that this result is based on the best automatically-generated lexicon (i.e., S-dict), instead of the best combined lexicon. Derczynski *et al.* (2013) observed modest improvements on NER for organisations and locations, despite the fact that the IV lexicon is based on `Aspell` and many named entities (e.g., person names) are not included. Jabeen *et al.* (2013) obtained similar positive results on NER for a tweet summarisation system. Baldwin *et al.* (2013) measured the degree of lexical variation over a range of social media datasets, and found that noise is greatest in Twitter and YouTube comments.

As for downstream applications, Shalev (2013) analysed the impact of our combined normalisation lexicon in the context of Twitter First Story Detection (FSD) (Petrović *et al.* 2010; Petrović *et al.* 2012). The impact is evaluated by comparing the number of event-relevant tweets and the reporting time of the first story when using raw tweets and normalised tweets, respectively. The experiments suggest applying our lexicon in tweet preprocessing is able to get the earliest reporting time in their FSD task, although the normalisation doesn't deliver more event-related tweets.

Beyond evaluations using our off-the-shelf lexicon, some other work also demonstrated the effectiveness of normalisation using in-house modules. For instance, Zhang *et al.* (2013) demonstrated that text normalisation improves syntactic parsing accuracy. Wang and Ng (2013) reported text normalisation increases absolute BLEU score by approximately 1.4% in English and Chinese machine translation tasks. Additionally, some normalisation modules have been ported to in-domain NLP tools for tweets. For instance, Owoputi *et al.* (2013) applied Brown clustering (Brown *et al.* 1992) to group lexical variants with the canonical forms into a set of flat clusters such as $just, jus, \ldots, juss$.[42] This cluster module reduces the lexical variance in the data similarly to text normalisation, and consequently improves the POS tagging accuracy.

[42]These flat clusters are truncated from the original hierarchical cluster.

3.7 Summary

In this chapter, we have proposed the task of normalising OOV non-standard words (i.e., lexical variants) to their canonical forms for short text messages in social media such as Twitter. We first analysed in-domain OOV word types and the distribution of each, and then proposed a candidate generation-and-selection normalisation approach using both contextual and string similarity information. The proposed method generally outperformed other benchmarks in a token-based normalisation task setting, however, the proposed method and many other benchmarks suffer from poor performance at lexical variant detection, which makes them less practical for real-world normalisation.

Encouraged by the performance of lexicon-based normalisation (i.e., an Internet slang dictionary), we moved on to using contextual/string similarity information to build a pure type-based normalisation lexicon with a particular focus on context-insensitive lexical variants (with length ≥ 4). Although the proposed type-based method has the limitation that it cannot capture context or disambiguate different usages of the same token. In empirical evaluation, we showed it to achieve state-of-the-art results at the time of publication, when combined with existing lexicons. The combined lexicon has broader coverage than existing dictionaries and reasonable precision. This type-based approach integrates the detection and normalisation of lexical variants into a simple, lightweight solution which is suitable for processing high-volume tweet feeds.

We further extended our evaluation of type-based approach to a downstream Twitter POS tagging task. The results suggest normalisation helps in boosting POS tagging accuracy, although the accuracy of using text normalisation is outperformed by a dedicated in-domain Twitter POS tagger.

In addition to experimenting on English data, the generality of a type-based approach was also demonstrated over Spanish text normalisation. Our experiments on the Spanish text normalisation once again demonstrated that the automatically-derived lexicon complements existing Spanish Internet slang lexicons.

Finally, we summarised recent progress on text normalisation, including recently

developed methods and recent results of applying text normalisation to other downstream NLP tasks and applications. The positive results reported by other researchers using our combined lexicon suggest the general effectiveness of text normalisation.

Overall, the exploration of text normalisation suggests it is a challenging task and has many challenges including processing efficiency, but the effort of developing suitable text normalisation methods is not in vain. Experiments on a number of downstream NLP tasks and applications indicate the importance of this text processing task.

Chapter 4

Geolocation Prediction

This chapter investigates assigning geospatial information to social media data. In particular, text-based methods are explored and improved to predict a Twitter user's primary location from a discrete set of pre-defined geographical entities, e.g., cities. We propose a unified geolocation framework to incorporate a range of factors in the geolocation prediction such as feature sets, data size, tweeting language and user metadata. These factors are examined to reveal their impact on the overall geolocation prediction accuracy. We also provide a detailed discussion on feature selection methods and relevant benchmarks in addition to the geolocation prediction literature in Chapter 2. Furthermore, we analyse user geolocatability and prediction confidence to calibrate the prediction accuracy for practitioners.

4.1 Introduction

The growing volume of user-generated text posted to social media services such as Twitter, Facebook, and Tumblr can be leveraged for many purposes ranging from natural disaster response to targeted advertising (Tuten 2008; Núnez-Redó *et al.* 2011; Yin *et al.* 2012). In many circumstances it is important to know a user's location in order to accomplish these tasks effectively. For example, disaster response managers must know where to direct resources in order to effectively coordinate aid, and advertisers could benefit from tailoring advertisements to a user's location. Similarly,

search results localisation hinges on knowledge of a user's location. Although many social media services allow a user to declare their location, such metadata is known to be unstructured and ad hoc (Hecht *et al.* 2011) (e.g., *melbo* denoting *Melbourne, AU*[1]), as well as oftentimes non-geographical (e.g., *in my own little bubble*). Text-based geolocation — automatically predicting a user's location based on the content of their messages — is therefore becoming of increasing interest (Cheng *et al.* 2010). In this chapter we investigate and improve text-based geolocation prediction for Twitter users. Specifically, we exploit the tweets and profile information of a given user to infer their primary city-level location, which we claim is sufficiently fine-grained to support the sorts of applications mentioned above.

As is well established in the literature (and discussed in Section 2.3.3 on Page 57), word choices and topics in social media differ across regions and can be used to infer geolocations. For example, a user in London is much more likely to talk about *Piccadilly* and *tube* than a user in New York or Beijing. That is not to say that those words are uniquely associated with London, of course: *tube* could certainly be mentioned by a user outside of the UK. However, the use of a range of such words with high relative frequency is strongly indicative of the fact that a user is located in London. Most work in this area utilises geotagged data as ground truth for evaluation (Eisenstein *et al.* 2010). The geotagged data contains GPS coordinates inserted with the user's consent by a GPS-enabled device such as a smartphone, and offers accurate information about a user's position at the time of tweeting.

The proposed text-based method primarily uses words for geolocation prediction, and intentionally excludes Twitter specific entities, such as hashtags and user mentions. The prediction accuracy therefore largely depends on whether the text contains sufficient geospatial information for geolocation prediction. Therefore, although this chapter focuses exclusively on Twitter, the proposed method could equally be applied to other forms of social media text, such as Facebook status updates or user-submitted comments (to services such as YouTube).

[1]Throughout the chapter, we present city names with ISO 3166-1 alpha-2 country-level designators such as *AU* = Australia and *CA* = Canada. Where US-based city names are mentioned in the context of the North American regional dataset used in experimentation (NA), we use an ISO 3166-2:US designator such as *US-CA* = California or *US-PA* = Pennsylvania.

Although approaches to text-based geolocation are offering increasingly promising results, the studies to date on this topic have been limited in a number of important ways. In the rest of the chapter, we address each of the following issues in turn.

Location Indicative Words. Given that text-based methods rely on salient words local to particular regions to disambiguate geolocations, do "location indicative words" improve the accuracy over using the full word set? Text-based geolocation prediction models for social media are predominantly based on the full text data of tweets, including common words with no geospatial dimension (e.g., *today*), potentially hampering prediction, and because of the large number of words observed in tweets, leading to slower, more memory-intensive models. We tackle this by automatically finding location indicative words (LIWs) via feature selection, and demonstrating the impact of the reduced feature set in geolocation prediction in Sections 4.4 and 4.5, corresponding to regional and global datasets, respectively.

Non-geotagged Tweets. In addition to experimenting with geotagged data, we further extend our analysis to incorporate non-geotagged tweets. Some recent work (Roller *et al.* 2012) has incorporated non-geotagged training data, although little work has analysed the contribution of non-geotagged data, i.e., the extent to which incorporating non-geotagged data improves geolocation accuracy. Furthermore, the evaluation of previous models has been restricted to geotagged data (in order to have access to a ground truth) although the goal of this line of research is to be able to infer locations for users whose locations are not known. However, it is unclear how well models evaluated only on geotagged data will generalise to non-geotagged data. For example, because geotagged tweets are sent from GPS-enabled devices such as smartphones, while non-geotagged tweets are sent from a range of devices (including desktop computers), these two types of data could have different characteristics (Gouws *et al.* 2011b).

Relative to this background, our explorations focus on following questions: Does a model trained on geotagged data generalise to non-geotagged data? What is the impact of adding non-geotagged texts to the training and test data? Is there an inherent sub-domain difference between geotagged and non-geotagged tweets given that geotagged tweets are primarily sent from mobile devices? In Section 4.6, we address

these issues by training and testing on geotagged tweets, non-geotagged tweets, and the combination of the two.

Language Influence. With some exceptions (Kinsella *et al.* 2011), most text-based geolocation studies have been carried out in an English-only setting, or a primarily English setting. Because high-accuracy language identification tools (Lui and Baldwin 2012) are now readily available, this is not a problem: messages in the target language can be identified, and text-based geolocation methods can be applied to only those messages. However, it remains to be seen whether text-based geolocation approaches that have been shown to work well for English perform as well on other languages, or perform well in a multilingual setting. English is tweeted throughout the world, whereas languages such as Indonesian are primarily tweeted in localised areas. As such, the performance of methods developed and tested over English data could be very different when applied to other languages. Furthermore, if language does influence accuracy, how can we exploit this to improve multilingual geolocation prediction? In Section 4.7, we investigate the language influence on a multilingual dataset.

Metadata and Ensemble Learning. Although tweet-based geolocation is worthy of study in its own right, tweets are accompanied by rich metadata in public user profiles.[2] While there has been some work on utilising timezone (Mahmud *et al.* 2012) and user-declared location (Hecht *et al.* 2011) information for user geolocation, the metadata remains largely untouched in the literature. Does the user-declared text metadata provide geographical information complementary to that in the tweets themselves? How can we make use of these multiple sources of textual data to produce a more accurate geolocation predictor? In Section 4.8, we address these questions by investigating the performance of metadata-based geolocation models and comparing them with benchmark methods.

Temporal Influence. Because Twitter is a growing and evolving medium, the data in Twitter streams tends to be locally temporal to the time of posting. In addition to evaluating the geolocation model on "old" time-homogeneous data (sampled

[2]The goal of this exploration is to improve the geolocation prediction accuracy by adding more Twitter specific features.

from the same time period as the training data), in Section 4.9 we evaluate the trained model on a "new" time-heterogeneous dataset to examine the temporal factor influence on the model generalisation, i.e., will a model trained on "old" data perform comparably on "new" test data?

User Geolocatability. We further discuss the geolocatability of users with regard to tweeting behaviour in Section 4.10. For instance, does mentioning many local place names have a strong influence on the prediction accuracy? Are there steps a user can take to reduce the risk of inadvertently leaking geographical information while sharing tweets with the public?

Prediction Confidence. Because of different tweeting behaviours among users, not all users are equally geolocatable, with only predictions for a proportion of them being reliable. A corresponding question is can measures of prediction confidence be formulated to estimate the accuracy of the geolocation prediction? We conduct a pilot study on approximating the prediction confidence through a range of variables in Section 4.11.

4.2 Geolocation Prediction Framework

In this chapter, we focus on predicting Twitter users' primary (referred to as their "home") location, and following (Cheng *et al.* 2010) and others, assume that a given user will be based in a single city-based location throughout the time period of study. We approach geolocation prediction as a text classification task. Tweets from each city are taken to represent a class. All tweets from a given user are aggregated and assigned to that user's primary location. We characterise geolocation prediction by four key components, which we discuss in turn below: (1) the representation of different geolocations, (2) the model, (3) the feature set, and (4) the data.

4.2.1 Representation: Earth Grid vs. City

Geolocations can be captured as points, or clustered based on a grid (Wing and Baldridge 2011; Roller *et al.* 2012), city centres (Cheng *et al.* 2010; Kinsella *et al.*

2011) or topic regions (Eisenstein *et al.* 2010; Hong *et al.* 2012). A point-based representation presents computational challenges, and is too fine-grained for standard classification methods. As for dynamic location partitioning, the granularity of regions is hard to control and will potentially vary across time, and the number of regions is a variable which will depend on the dataset and potentially also vary across time. Fixed grid-based representations are hindered because there is considerable variability in the shape and size of geographical regions: a coarse-grained grid cell is perhaps appropriate in central Siberia, but for densely-populated and linguistically/culturally diverse regions such as Luxembourg, doesn't lead to a natural representation of the administrative, population-based or language boundaries in the region. We therefore opt for a city-based representation, which is able to capture these boundaries more intuitively. The downside to this representation is that it is inappropriate for classifying users in rural areas. As we will see in Figure 4.1, however, the bulk of Twitter users are, unsurprisingly, based in cities.

We use the publicly-available `Geonames` dataset as the basis for our city-level classes.[3] This dataset contains city-level metadata, including the full city name, population, latitude and longitude. Each city is associated with hierarchical regional information, such as the state and country it is based in, so that London, GB, e.g., is distinguished from London, CA. We hence use a city-region-country format to represent each city (e.g., Toronto, CA is represented as toronto-08-ca, where 08 signifies the province of Ontario and ca signifies Canada).[4] Because region coding schemes vary across countries, we only employ the first- and second-level region fields in `Geonames` as the region. Furthermore, if the second-level field is too specific (i.e., longer than 4 letters in our setting), we only incorporate the first-level region field (e.g., instead of using melbourne-07-24600-au, we use melbourne-07-au). Moreover, because cities are sometimes complex in structure (e.g., Boston, US colloquially refers to the metropolitan area rather than the city, which is made up of cities including Boston, Revere and Chelsea), we collapse together cities which are adjacent to one another within a

[3] `http://www.geonames.org` (Retrieved 25/10/2012)

[4] Country code information can be found in `http://download.geonames.org/export/dump/countryInfo.txt`

single administrative region, as follows:

1. Identify all cities which share the same region code (i.e., are located in the same state, province, county, etc.) in the `Geonames` dataset.
2. For each region, find the city c with the highest population.
3. Collapse all cities within 50km of c into c.[5]
4. Select the next-largest city c, and repeat.
5. Remove all cities with a population of less than 100K. The remaining cities form our city-based representation of geolocations.

As a result of this methodology, Boston, US ends up as a single city (incorporating Revere and Chelsea), but neighbouring Manchester, US is a discrete city (incorporating Bedford) because it is in New Hampshire. This algorithm identifies a total of 3,709 collapsed cities throughout the world.

4.2.2 Geolocation Prediction Models

Various machine learning algorithms can be applied to the task of multi-class text categorisation. However, many state-of-the-art learning algorithms are not appropriate for this particular task for reasons of scalability. For example, support vector machines are not well suited to massively multi-class problems (i.e., 3,709 cities in our case). Finally, we would ideally like to have a learning algorithm which can be easily retrained, e.g., to incorporate new training data from the Twitter data stream. As such, we primarily experiment with simple learning algorithms and ensemble learning for geolocation prediction.

[5]We use the great-circle distance (Vincenty 1975) for all distance calculations in our experiments, as opposed to Euclidean distance, to properly capture the three-dimensional surface of the earth. The proximity of cities varies across the world, e.g., cities on the east coast of the United States are much closer to each other than major cities in Australia. There is therefore scope to explore the impact of this 50km setting on the city label set, which we leave to future work.

Generative vs. Discriminative Models

As discussed in literature (in Section 2.3.3 on Page 60), generative models (e.g., naive Bayes) are based on estimation of joint probability of observing a word vector and a class (i.e., $P(w_1, w_2, \ldots, w_n, c_i)$, where $w_1, w_2, \ldots$ are words and $c_i \in C$ is a city from a combined set of cities C). In contrast, discriminative models are based on estimation of a class given a word vector (i.e., $P(c|w_1, w_2, \ldots, w_n)$). The objective of both models is to find a city $c_{max} \in C$ such that the relevant probability is maximised. In our experiments, we use both types of models. For instance, we choose a state-of-the-art discriminative geolocation model based on KL divergence over k-d tree partitioned unigrams (KL) (Roller *et al.* 2012). We also adopt a generative multinomial naive Bayes (NB) model (Hecht *et al.* 2011) as our default benchmark, for it incorporates a class prior, allowing it to classify an instance in the absence of any features shared with the training data.

Single vs. Ensemble Models

In addition to single model comparisons (e.g., discriminative KL versus generative NB in Sections 4.4 and 4.5), we further combine multiple base classifiers — e.g., heterogeneous NB models trained on each of Twitter text and user metadata — to improve the accuracy. First, we investigate the accuracies of base classifiers and correlations between them. Then, we apply different ensemble learning strategies in Section 4.8.

4.2.3 Feature Set

Predominantly, geolocations are inferred based on geographical references in the text, e.g., place names, local topics or dialectal words. However, these references are often buried in noisy tweet text, in which lexical variants (e.g., *tmrw* for "tomorrow") and common words without any geospatial dimension (e.g., *weather*, *twitter*) are prevalent. These noisy words have the potential to mislead the model and also slow down the processing speed. To tackle this issue, we perform feature selection to identify "location indicative words". Rather than engineering new features or attempting

to capture named entities (e.g., *the White House*) or higher-order n-grams, we focus on feature selection over simple word unigrams (see Section 4.3). This is partly a pragmatic consideration, in that unigram tokenisation is simpler.[6] Partly, however, it is for comparability with past work, in determining whether a strategically-selected subset of words can lead to significant gains in prediction accuracy (see Sections 4.4 and 4.5).

In addition to feature selection, the feature set can be further refined and extended in various ways. For instance, feature selection can be enhanced by incorporating non-geotagged tweet data. Furthermore, languages can be used to shape the feature set, as words from different languages carry varying amounts of geospatial information, e.g., because Dutch is primarily used only in the Netherlands, Dutch words are usually more location indicative than English words. Moreover, user-provided metadata (e.g., location and timezone) is readily accessible in the tweet JSON objects. This metadata can be appended as extra text features, in addition to features derived from tweet text. We investigate the impact of these factors in later sections.

The exploration of the feature set is aimed at geolocation prediction accuracy by distinguishing between different features. While feature selection as a preprocessing procedure is one way to achieve this, there also exist other options such as incorporating regularisation when training models. These explorations aim at boosting the prediction accuracy in different ways. Feature selection imposes a hard cut-off on the ranked features, and the learner operates on the reduced feature set. In contrast, regularisation incorporates all features but weights them accordingly, including assigning irrelevant features with tiny weights. These features contribute little to the prediction. The different weights suggest the importance of features, which can be considered as feature prioritisation. Although regularisation may improve the accuracy, feature selection is more appropriate for the task, due to computational tractability in training and (to a lesser extent) at runtime. Furthermore, finding an appropriate regularisation function is non-trivial. It is possible that commonly-used regularisers (e.g., L1 and L2 regularisation) are inappropriate.

[6] Also, preliminary results with both named entities and higher order n-grams were disappointing.

4.2.4 Data

Geolocation prediction models have primarily been trained and tested on geotagged data.[7] We use both regional datasets (i.e., geotagged tweets collected from the continental US: Eisenstein *et al.* (2010); Mahmud *et al.* (2012)) and global datasets (Kinsella *et al.* 2011) in this research. Because of accessibility issues (e.g., many tweets in older datasets have been deleted and are thus not accessible now) and data sparseness (e.g., there were only 10K users in the study of (Eisenstein *et al.* 2010)), we are only able to experiment over a small number of public datasets. In this chapter, we employ three geotagged datasets:

1. A regional North American geolocation dataset from Roller *et al.* (2012) (NA hereafter), for benchmarking purposes. NA contains 500K users (38M tweets) from a total of 378 of our pre-defined cities. NA is used as-is to ensure comparability with previous work in Section 4.4.

2. A dataset with global coverage (WORLD hereafter), collected via the Twitter public Streaming API from 21/09/2011 to 29/02/2012. The tweet collection is further shaped for different evaluation tasks, e.g., geotagged English data WORLD in Section 4.5, incorporating non-geotagged English data WORLD+NG in Section 4.6, multilingual geotagged data WORLD+ML in Section 4.7 and with rich metadata WORLD+META in Section 4.8.

3. A second dataset with global coverage novel to this research (LIVE), which contains tweets collected more than 1 year after WORLD (from 03/03/2013 to 03/05/2013), to analyse the influence of temporal recency on geolocation prediction. Unlike the other two datasets, LIVE is used only as a test dataset, in Section 4.9.

WORLD was restricted to English tweets in order to create a dataset similar to NA (in which English is the predominant language), but covering the entire world.

[7] One exception to this is Cheng *et al.* (2010), who train on users whose user-declared metadata location fields correspond to canonical locations (e.g., Boston, MA), and test on users whose locations are indicated with GPS coordinates in their metadata.

Filtering criterion	Proportion of tweets (relative to preceding step)
Geotagged	0.008
Near a city	0.921
Non-duplicate and non-Foursquare	0.888
English	0.513

Table 4.1: Proportion of tweets remaining after filtering the data based on a series of cascaded criteria. These numbers are based on a Twitter corpus collected over two months.

It was preprocessed by filtering the data as follows. First, all non-geotagged tweets were removed. Next, we eliminated all tweets that aren't close to a city by dividing the earth into $0.5^\circ \times 0.5^\circ$ grid cells, and discarding any tweet for which no city in our `Geonames` class set is found in any of the 8 neighbouring grid cells. We then assign each user to the single city in which the majority of their tweets occur. We further remove cities with fewer than 50 feature types (i.e., word types) to reduce data sparsity. This results in 3135 cities in WORLD (as opposed to 3709 cities in the full `Geonames` class set). We eliminated exact duplicate tweets and Foursquare check-ins (which encode the user location in the form of *I'm at ...*). After that, non-English tweets were further removed using `langid-2012`, an open-source language identification tool (Lui and Baldwin 2012). This filtering is summarised in Table 4.1 which also shows the proportion of tweets remaining after each step. The total number of users and tweets in WORLD is 1.4M and 12M, respectively. Similar to NA, the development and test datasets both contain 10K users, and the remainder of the users are used in training. The development and test data was sampled such that each user has at least 10 geotagged tweets to alleviate data sparsity.[8] We tokenised the tweets with a Twitter-specific tokeniser (adapted from O'Connor *et al.* (2010)).

[8]This restriction was not applied to the training data.

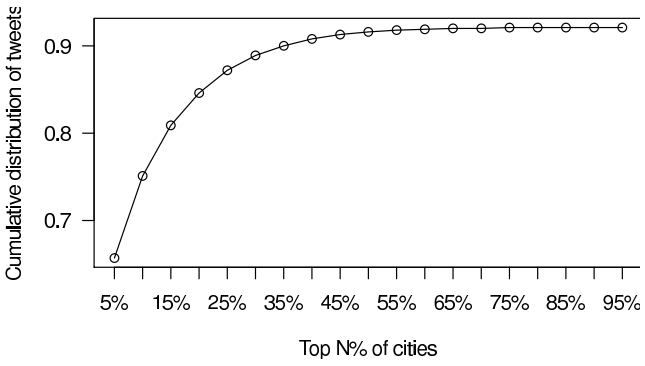


Figure 4.1: Cumulative coverage of tweets for increasing numbers of cities based on 26 million geotagged tweets.

Although there are certainly instances of social media users with high mobility (Li *et al.* 2012a), recent studies have shown that most users tend to tweet from within a limited region (Cho *et al.* 2011; Hecht *et al.* 2011). We also analyse the spread of WORLD in Figure 4.2, in terms of: (1) the number of users with at least 10 geotagged tweets; and (2) the number of users with differing levels of geographical spread in their tweets, measured as the average distance between each of a user's tweets and the centre of the city to which that user is allocated.[9] This preliminary analysis shows that most users have a relatively small number of geotagged tweets, and most users stay near a single city (e.g., 83% users have a geographical spread of 50 kilometres or less). The high proportion of users with an average distance of 1km to the city centre is an artefact of their geotagged tweets being mapped to a city centre before performing this analysis. In order to investigate the coverage of the proposed city-based partition, we examine the recall in our original sample of 26 million geotagged tweets (prior to filtering, as described above). The analysis reveals that 92.1% of tweets are "close" to (in a neighbouring $0.5° \times 0.5°$ grid cell) to one of our pre-defined cities, and that the top 40% of cities contain 90% of the geotagged

[9]The geographical spread is calculated over a random sub-sample of 10 tweets for a given user, for efficiency reasons.

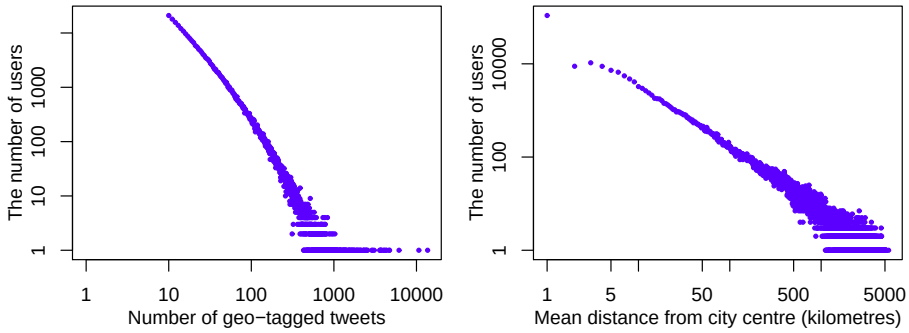


Figure 4.2: The number of users with different numbers of tweets, and different mean distances from the city center, for WORLD.

tweets after filtering, as shown in Figure 4.1. This supports our assumption that most (geotagged) Twitter users are based in cities.

4.2.5 Evaluation Measures

Having formulated the geolocation prediction task into a discrete class space through the use of our city class set, it is possible to use simple classification accuracy to evaluate our models. However, given that all of our class labels have a location (in the form of latitude–longitude coordinates), we can also sensitise the evaluation to distance-based predictive error. For instance, if the correct location for a user is Seattle, US, a prediction of Portland, US is arguably better than a prediction of Los Angeles, US, on the basis of geospatial proximity. We use a number of evaluation measures which capture spatial proximity, in line with previous work (Wing and Baldridge 2011; Roller *et al.* 2012):[10]

1. **Acc**: city-level accuracy, i.e., the proportion of predictions that correspond to

[10]In very recent work, Priedhorsky *et al.* (2014) additionally proposed a set of probabilistic metrics to evaluate tweet-based geolocation prediction, including using the expected distance between a tweet's true point location to a random point location drawn from the probability distribution of the geolocation model. While we strongly support this new direction for geolocation modelling and evaluation, depending on the application context, we argue that point- or region-based representations and related discrete evaluation measures are equally important in user geolocation research.

the correct city;

2. **Acc@161**: the proportion of predictions that are within a distance of 161 kilometres (100 miles) from the correct city-level location. This empirical measure (Cheng *et al.* 2010) is a relaxed version of Acc, capturing near-miss predictions.

3. **Acc@C**: country-level accuracy, i.e., the proportion of predicted locations that are in the same country as their corresponding true locations. This measure is useful for applications relying on country-specific Twitter data, e.g., sentiment analysis in specific countries.

4. **Median**: median prediction error, measured in kilometres between the predicted city centres and the true geolocations. We prefer to use the median, as opposed to mean, distance because the median is less sensitive to wildly incorrect predictions — e.g., a user from London, GB classified as being based in Sydney, AU. In contrast, the mean distance can increase substantially due to a small number of extreme misclassifications, although this effect is limited for inherently-bounded regional datasets such as NA.

4.3 Finding Location Indicative Words

Precise user locations for individual messages are embedded in geotagged tweets in the form of latitude–longitude coordinates. By mapping these coordinates to cities and representing each tweet as a bag of words, we are able to make connections between words (i.e., features) and cities (i.e., classes). Having set up this connection, we are able to apply a range of feature selection methods to identify useful features in geolocation prediction. Next, we first briefly review the literature of feature selection, and then discuss recent feature selection approaches adopted for the Twitter geolocation prediction task.

4.3.1 Literature on Feature Selection

Feature selection studies how to choose features to fill in a machine learning algorithm for better accuracy. They are generally categorised into three types: *filters*, *wrappers*, and *embedded* methods (Guyon and Elisseeff 2003).[11]

A *filter* calculates the usefulness of each feature with respect to a pre-defined metric, e.g., information gain. Usually, the score of usefulness for each feature is calculated individually, and is independent of the machine learning algorithm that the selected feature set will be deployed over. As a result, a *filter* eventually delivers a feature ranking relative to the metric, rather than an optimal set of features for the machine learning algorithm. Cross validation or held-out development data is often then used to obtain the n-best features. Note that the n-best features ranked by a *filter* may not be the optimal feature set or even not the best-n features for the machine learning task. This is mainly due to two reasons: First, because of feature interactions, some features are less useful when they are treated alone, however, they may be useful when combined together (John *et al.* 1994), e.g., *tube* and *BBC* are indicative of London in UK. Second, a *filter* method indicates the feature usefulness by its own metric (i.e., is subject to inductive bias) which may sometimes be substantially different to that of the learner, that is, the good features considered by a *filter* may be bad features for the machine learning algorithm on the task.

A *wrapper* measures the usefulness of a feature set in a more straightforward way. Given a particular feature set, a *wrapper* trains a model using the same machine learning method as will be used in the deployed model, and then evaluates the model on held-out development data. The accuracy of the model is then adopted as a measure of usefulness for the given feature set. It is often impossible to apply a *wrapper* as a practical feature selection strategy when the number of features is large. This is primarily because model training and evaluation will be performed for a large number of feature sets, which is time-consuming in general.[12] In addition, even if

[11]One can certainly construct various new features out of the original features ("feature construction"), and then put the constructed features into the same selection process. Due to efficiency reasons, this thesis primarily focuses on feature selection in which a subset of original features is selected from the original input features without further considering the potential feature interactions.

[12]Theoretically, it requires exponential complexity (i.e., 2^k if k is the feature number) to guarantee

the number of feature sets is small, a *wrapper* may still be throttled by the model training efficiency, i.e., training a model is inefficient and time-consuming.

Additionally, some methods have built-in feature selection functionality, denoted as *embedded* methods. A typical *embedded* method combines two parts when choosing features: model fitness for the data and model complexity. On the one hand, a good model requires a larger number of parameterised features to characterise the data. On the other hand, a large parameter number is undesirable, as too many parameters often result in overfitting to the data, in particular, when the feature number is larger than that of training instances. Given this paradox on parameterised feature numbers, these two parts compete against each other to select the optimal feature set.

Having analysed different approaches for feature selection, we now turn to discuss how to select words encoding geospatial information in Twitter geolocation prediction. A massive amount of data can be harvested from the Twitter Streaming API. Furthermore, in Twitter data, non-standard words and named entities are prevalent, often resulting in a large number of word types. As a result, we opt for a filter-based approach to feature selection due to its simplicity and efficiency.

4.3.2 Location Indicative Words

In this section, we present a range of methods for ranking these words by their location indicativeness, i.e., the degree to which a word is associated with particular cities. Words that either explicitly (e.g., place names) or implicitly (e.g., dialectal words, slang or local references) encode geographical information are collectively referred to as "location indicative words" (LIWs); it is these words that we aim to automatically identify. Examples of LIWs are:

1. local words (1-local) that are used primarily in a single city, namely *yinz* (used in Pittsburgh to refer to the second-person plural pronoun), *dippy* (used in Pittsburgh to refer to a style of fried egg, or something that can be dipped

the selection of optimal feature set. In practise, heuristics such as Akaike information criterion (AIC) or greedy selection/elimination strategies are often used to reduce the complexity (Kohavi and John 1997).

in coffee) and *hoagie* (used primarily in Philadelphia, to refer to a kind of sandwich);[13]

2. semi-local words (n-local) that refer to some feature of a relatively limited subset of cities, namely *ferry* (found, e.g., in Seattle, New York and Sydney), *Chinatown* (common in many of the largest cities in the US, Canada and Australia, but much less common in European and Asian cities), and *tram* (found, e.g., in Vienna, Melbourne and Prague)

In addition to LIWs there are common words (common) which aren't expected to have substantial regional frequency variation, namely *twitter*, *iphone* and *today*.

In the remainder of this section, we present various feature selection methods for identifying LIWs (Chang *et al.* 2012; Laere *et al.* 2013b). Many are well-known methods (e.g., *IGR*), however, we also tried new methods inspired by geospatial factors such as *Ripley*, *GeoSpread* and *GeoDen*. The feature selection methods can be broadly categorised into three types: (1) statistical; (2) information-theoretic; and (3) heuristic. To reduce low-utility words and noise, for all feature selection methods, we remove all words which include non-alphabetic letters, are less than 3 letters long, or have a word frequency < 10.

4.3.3 Statistical-based Methods

Statistical hypothesis testing is often used to determine whether an event occurs by chance (i.e., the null hypothesis) or not (i.e., the alternative hypothesis) at a particular confidence level (e.g., $95\% \equiv p < 0.05$). In our case, an event is defined to be a co-occurrence between a word and a city, and the null hypothesis assumes the co-occurrence is by chance, i.e., the word and city are independent. The goal of feature selection is then to find word–city pairs where the null hypothesis is rejected.

[13]These words were identified with the aid of datasets of regional words such as DARE: `http://dare.wisc.edu/`.

	in c	not in c
w	$O_{w,c}$	$O_{w,\bar{c}}$
non-w word	$O_{\bar{w},c}$	$O_{\bar{w},\bar{c}}$

Table 4.2: Contingency table for word and city co-occurrence.

χ^2 and Log-Likelihood

The χ^2 statistic is commonly used to examine the degree of independence between random variables. A contingency table representing the observations of the variables is formed, as in Table 4.2. The general form of the statistic is:

$$\sum_{i}^{n} \frac{(O_i - E_i)^2}{E_i}$$

where O_i represents an observation (i.e., co-occurrence of a city (c) and word (w)), and n is the number of cells in the table. $O_{w,c}$ and $O_{\bar{w},\bar{c}}$ denote the occurrence of word w in city c and non-w words in cities other than c, respectively. $E_{w,c}$ denotes the expected frequency of w in c, calculated from the marginal probabilities and total counts N:

$$\begin{aligned} E_{w,c} &= P(w) \times P(c) \times N = \frac{O_{w,c} + O_{w,\bar{c}}}{N} \times \frac{O_{w,c} + O_{\bar{w},c}}{N} \times N \\ N &= O_{w,c} + O_{\bar{w},c} + O_{w,\bar{c}} + O_{\bar{w},\bar{c}} \end{aligned}$$

If the χ^2 statistic is larger than the number in the χ^2 distribution, with respect to the degrees of freedom (in this case, 1), then the null hypothesis that city c and word w are independent is rejected. As with many statistical tests, χ^2 can be ineffective when counts are low. We address this through our word frequency thresholding and use of massive amounts of training data.

Conventionally, χ^2 is used to identify the set of features which satisfies a pre-defined confidence level (e.g., $p < 0.05$). However, in the case of LIW selection, we instead use the χ^2 statistic to rank all word–city pairs. The selection of LIWs is deferred to the parameter tuning state, in which the boundary between LIWs and common words is optimised using development data.

At this point, a different ranking of LIWs is produced per city, where what we desire is a global ranking of LIWs capturing their ability to discriminate between cities in the combined label set. There are various ways to do this aggregation. As suggested by Laere *et al.* (2013b), one approach to selecting n features based on χ^2 is to iteratively aggregate the top-m features from each city until n features are obtained. Alternatively, they can be ranked based on the highest-scoring occurrence of a given word for any city, by first sorting all city–word χ^2 test pairs, then selecting the first occurrence of a word type for the aggregated ranking. These two aggregation approaches produce different feature selection rankings, and are distinguished using *Chi* and *MaxChi*, respectively.[14]

Similar to the χ^2 test, the log-likelihood ratio ("*Loglike*": Dunning (1993)) has also been applied to LIW selection (Laere *et al.* 2013b). The *Loglike* test determines whether h_0 (the null hypothesis, i.e., the word is independent of the city) is more likely than h_1 (the alternative hypothesis, i.e., the word is dependent on the city). Following Dunning (1993), the likelihood of a hypothesis, $L(\cdot)$, is estimated using binomial distributions.

$$L(h_1) = p_1^{k_1}(1-p_1)^{n_1-k_1}\binom{n_1}{k_1}p_2^{k_2}(1-p_2)^{n_2-k_2}\binom{n_2}{k_2}$$

$$p_1 = P(w|c) = \frac{k_1}{n_1} = \frac{O_{w,c}}{O_{w,c} + O_{\bar{w},c}}$$

$$p_2 = P(w|\bar{c}) = \frac{k_2}{n_2} = \frac{O_{w,\bar{c}}}{O_{w,\bar{c}} + O_{\bar{w},\bar{c}}}$$

k_1 (k_2) represents the occurrences of word w in city c (not in city c), and n_1 (n_2) represents all word occurrences in city c (not in city c). $L(h_0)$ is a special case of $L(h_1)$ for which p_1 and p_2 are equal, as below:

$$p_1 = p_2 = p = \frac{O_{w,c} + O_{w,\bar{c}}}{N}$$

[14]One possible alternative to computing χ^2 for each word and city, and then aggregating these values into a final ranking of words, would be to compute a single χ^2 value for each word from a contingency table with 2 rows as in Table 4.2, but with one column per city. Nevertheless, this is not the standard use of χ^2 in feature selection, and we leave this possibility to future work.

The *Loglike* test statistic is then expanded using observations:

$$\begin{aligned} Loglike(w) = 2[&O_{w,c}\log O_{w,c} + O_{\bar{w},c}\log O_{\bar{w},c} + O_{w,\bar{c}}\log O_{w,\bar{c}} + O_{\bar{w},\bar{c}}\log O_{\bar{w},\bar{c}} + N\log N \\ &- (O_{w,c} + O_{\bar{w},c})\log(O_{w,c} + O_{\bar{w},c}) - (O_{w,\bar{c}} + O_{\bar{w},\bar{c}})\log(O_{w,\bar{c}} + O_{\bar{w},\bar{c}}) \\ &- (O_{\bar{w},c} + O_{\bar{w},\bar{c}})\log(O_{\bar{w},c} + O_{\bar{w},\bar{c}}) - (O_{w,c} + O_{w,\bar{c}})\log(O_{w,c} + O_{w,\bar{c}})] \end{aligned}$$

Having calculated the *Loglike* for each word–city pair, we then aggregate across cities similarly to *Chi* (by selecting the top-m features per city until n features are obtained), following Laere *et al.* (2013b).[15]

Ripley's K Statistic

Spatial information can also be incorporated into the hypothesis testing. For example, the Ripley K function (*Ripley*: O'Sullivan and Unwin (2010)) measures whether a given set of points is generated from a homogeneous Poisson distribution. The test statistic calculates the number of point pairs within a given distance λ over the square of the total number of points. With regards to LIW selection, the set of points (Q_w) is the subset of geotagged users using a particular word w. The test statistic is formulated as follows (Laere *et al.* 2013a):

$$K(\lambda) \quad = \quad A \times \frac{|\{p, q \in Q_w : \text{distance}(p, q) \leq \lambda\}|}{|Q_w|^2}$$

where A represents the total area under consideration (e.g., the whole of North America, or the whole globe); this is dropped when generating a ranking.

A larger value of $K(\lambda)$ indicates greater geographical compactness of the set Q_w (i.e., p and q are spatially close). However, $|Q_w|$ (i.e., the number of users who use word w) varies considerably across words, and can dominate the overall statistic. A number of variations have been proposed to alleviate this effect, including replacing the denominator with a factor based on L1, and taking the logarithm of the overall value (Laere *et al.* 2013a). The quadratic computational complexity of *Ripley* becomes

[15]Note also that, as we will see later in our experiments, there is almost no empirical difference between the two aggregation methods for χ^2, so the choice of aggregation method here is largely arbitrary.

an issue when $|Q_w|$ is large (i.e., for common words). Randomised methods are usually adopted to tackle this issue, e.g., subsampling points from training data for *Ripley* calculation relative to different distances λ. For our experiments, we adopt the optimised implementation of Laere *et al.* (2013a) using $\lambda = 100$km with 5K samples.

4.3.4 Information Theory-based Methods

In addition to statistical methods, we also experiment with information-theoretic feature selection methods based on measures which have been shown to be effective in text classification tasks, e.g., Information Gain (IG) (Yang and Pedersen 1997).

Information Gain and Gain Ratio

Information Gain (IG) measures the decrease in class entropy a word brings about, where higher values indicate greater predictability on the basis of that feature. Given a set of words $\mathbf{w}$, the IG of a word $w \in \mathbf{w}$ across all cities ($\mathbf{c}$) is calculated as follows:

$$\begin{aligned} IG(w) &= H(\mathbf{c}) - H(\mathbf{c}|w) \\ &\propto -H(\mathbf{c}|w) \\ &\propto P(w)\sum_{c\in\mathbf{c}} P(c|w)logP(c|w) + P(\bar{w})\sum_{c\in\mathbf{c}} P(c|\bar{w})logP(c|\bar{w}) \end{aligned}$$

where $P(w)$ and $P(\bar{w})$ represent the probabilities of the presence and absence of word w, respectively. Because $H(\mathbf{c})$ is the same for all words, only $H(\mathbf{c}|w)$ — the conditional entropy given w — needs to be calculated to rank the features.

Words carry varying amounts of "intrinsic entropy", which is defined as:

$$IV(w) = -P(w)logP(w) - P(\bar{w})logP(\bar{w})$$

Local words occurring in a small number of cities often have a low intrinsic entropy, where non-local common words have a high intrinsic entropy (akin to inverse city frequency; see Section 4.3.5). For words with comparable IG values, words with smaller intrinsic entropies should be preferred. Therefore, following Quinlan (1993) we further normalise $IG(w)$ using the intrinsic entropy of word w, $IV(w)$, culminating

in information gain ratio (IGR):

$$IGR(w) = \frac{IG(w)}{IV(w)}$$

Logistic Regression-based Feature Weights

The previous two information-theoretic feature selection methods (IG and IGR) optimise across all classes simultaneously. Given that some LIWs may be strongly associated with certain locations, but are less tied to other locations, we also conduct per-class feature selection based on logistic regression (LR) modelling.[16] We consider this method to be information theoretic because of its maximisation of entropy in cases where there is uncertainty in the training data.

Given a collection of cities $\mathbf{c}$, the LR model calculates the probability of a user (e.g., represented by word sequence: $w_1, w_2, \ldots, w_n$) assigned to a city $c \in \mathbf{c}$ by linearly combining eligible LR feature weights:

$$P(c|w_1, w_2, \ldots, w_n) = \frac{1}{Z} exp(\sum_{k=1}^{m} \lambda_k f_k)$$

where Z is the normalisation factor, m is the total number of features, and f_k and λ_k are the features and feature weights, respectively. As with other discriminative models, it is possible to incorporate arbitrary features into LR, however, a feature (function) in our task is canonically defined as a word w_i and a city c: when w occurs in the set of messages for users in class c, a feature $f_k(w_i, c)$ is denoted as $[\text{class} = c \wedge w_i \in c]$. Each f_k maps to a feature weight denoted as $\lambda_k \in \mathcal{R}$. The method results in a per-city word ranking with words ranked in decreasing order of λ_k, from which we derive a combined feature ranking in the same manner as *MaxChi*.[17]

Notably, incorporating a regularisation factor balances model fitness and complexity, and could potentially achieve better results. We don't explicitly perform regularisation in the modelling stage. Instead, we first obtain the feature list ranked by LR as other feature selection methods and then evaluate the subset of top-n

[16]For the logistic regression modeller, we use the toolkit of Zhang Le (`https://github.com/lzhang10/maxent`), with 30 iterations of L-BFGS over the training data.

[17]As with *LogLike*, the choice of aggregation method here is largely arbitrary, based on our empirical results for χ^2.

ranked features on the development data. This is in fact equivalent to "filter-based" regularisation (cf. filter-based feature selection: (Guyon and Elisseeff 2003)).

Distribution Difference

LIW selection can be likened to finding words that are maximally dissimilar to stop words (Chang *et al.* 2012). Stop words like *the* and *today* are widely used across many cities, and thus exhibit a relatively flat distribution. In contrast, LIWs are predominantly used in particular areas, and are more skewed in distribution. To capture this intuition, LIW selection is then based on the "distribution difference" across cities between stop words and potential LIW candidates (i.e., all non-stop words). Given a pre-defined set of stop words S, the distribution difference is calculated as:

$$\mathit{DistDiff}(w_{ns}) = \sum_{w_s \in S} \mathit{Diff}(w_{ns}, w_s) \frac{\text{Count}(w_s)}{\text{Count}(S)}$$

where Count(w_s) and Count(S) denote the number of occurrences of a stop word w_s and the total number of occurrences of all stop words, respectively. The difference (i.e., $\mathit{Diff}(w_{ns}, w_s)$) between a stop word w_s and non-stop word w_{ns} can be evaluated in various ways, e.g., symmetric KL-divergence ($\mathit{DistDiff}_{skl}$), or the total variance ($\mathit{DistDiff}_{tv}$) of absolute probability difference across all cities $\mathbf{c}$ (Chang *et al.* 2012):

$$\begin{aligned} \mathit{Diff}_{skl}(w_{ns}, w_s) &= \sum_{c \in \mathbf{c}} P(c|w_{ns}) \log \frac{P(c|w_{ns})}{P(c|w_s)} + P(c|w_s) \log \frac{P(c|w_s)}{P(c|w_{ns})} \\ \mathit{Diff}_{tv}(w_{ns}, w_s) &= \sum_{c \in \mathbf{c}} |P(c|w_{ns}) - P(c|w_s)| \end{aligned}$$

where $P(c|w_{ns})$ and $P(c|w_s)$ denote the probability of a word occurring in a city in the per-word city distribution for w_{ns} and w_s, respectively. The non-stop words are then sorted by distribution difference in decreasing order. In our experiments, we use the implementation of Chang *et al.* (2012).

4.3.5 Heuristic-based Methods

Other than commonly-used feature selection methods, a number of heuristics can be used to select LIWs.

Decoupling City Frequency and Word Frequency

High-utility LIWs should have both of the following properties:

1. High Term Frequency (*TF*): there should be a reasonable expectation of observing it from the users' tweets in a city.

2. High Inverse City Frequency (*ICF*): the word should occur in tweets associated with a relatively small number of cities.

We calculate the *ICF* of a word w simply as:

$$icf_w = \frac{|\mathbf{c}|}{cf_w}$$

where $\mathbf{c}$ is the set of cities and cf_w is the number of cities with users who use w in the training data. Combining the two together, we are seeking words with high *TF-ICF*, analogous to seeking words with high *TF-IDF* values in information retrieval. In standard *TF-IDF* formulations, we multiply *TF* and *IDF*. A simple product of *TF* and *ICF* tends to be dominated by the *TF* component, however: for example, *twitter* scores as highly as *Jakarta*, because *twitter* has a very high *TF*. We resolve this by decoupling the two factors and applying a radix sort ranking: we first rank features by *ICF* then by *TF*, in decreasing order. As this approach is largely based on the inverse city frequency, we denote it as *ICF* below.

Geographical Spread and Density

LIWs have "peaky" geographical distributions (Cheng *et al.* 2010). In this section, we discuss two heuristic measures for LIW selection which are based on the geographical distribution of the word.

Geographical spread (*GeoSpread*: Laere *et al.* (2013b)) estimates the flatness of a word's distribution over cities. First, the earth is divided into 1° latitude by 1° longitude cells. For each word w, the cells in which w occurs are stored. Then, all neighbouring cells containing w are merged by multi-pass scanning until no more cells can be merged. The number of cells containing w after merging is further stored.

Finally, the *GeoSpread* score for the word w is calculated as follows:

$$GeoSpread(w) = \frac{\#\text{ of cells containing } w \text{ after merging}}{Max(w)}$$

where $Max(w)$ represents the maximum frequency of w in any of the original unmerged cells. Smaller values indicate greater location indicativeness. This measure was originally used to rank Flickr tags by locality, e.g., *London* is more location-indicative than *beautiful*. It ignores the influence of stop words, as they are not common in Flickr tags. However, stop words like *the* are frequent in Twitter, and occur in many locations, making the numerator small and denominator large. Furthermore, stop word frequencies in cells are usually high. Consequently, *the* has a similarly small *GeoSpread* to *London*, which is undesirable. In other words, *GeoSpread* is flawed in not being able to distinguish stop words from local words, although it can be effective at ranking less common words (e.g., *London* vs. *beautiful*).

Geographical density (*GeoDen*: Chang *et al.* (2012)) strategically selects peaky words occurring in dense areas. Given a subset of cities $\mathbf{c}' \subseteq \mathbf{c}$ where word $w \in \mathbf{w}$ is used, the *GeoDen* is calculated as:

$$\begin{aligned} GeoDen(w) &= \frac{\sum_{c \in \mathbf{c}'} P(c|w)}{|\mathbf{c}'|^2 \frac{\sum_{c_j, c_k \in \mathbf{c}' j \neq k} \text{dist}(c_j, c_k)}{|\mathbf{c}'|(|\mathbf{c}'|-1)}} \\ &= \frac{\sum_{c \in \mathbf{c}'} P(c|w)}{|\mathbf{c}'| \frac{\sum_{c_j, c_k \in \mathbf{c}' j \neq k} \text{dist}(c_j, c_k)}{|\mathbf{c}'|-1}} \end{aligned}$$

where $\text{dist}(c_j, c_k)$ is the great-circle distance between cities c_j and c_k. Similarly, $P(c|w)$ denotes the distribution of word w across each city $c \in \mathbf{c}'$. The denominator is made up of the square of the number of cities $|\mathbf{c}'|$ that w occurs in (which has a similar effect to *ICF* above), and the average distance between all cities where w is used. LIWs generally have a skewed geographical distribution in a small number of locations, meaning that the denominator is small and the numerator is large. The issue with this measure is the computational complexity for common words that occur in many cities. Furthermore, cities containing a small number of occurrences of w should not be incorporated, to avoid systematic noise, e.g., from travellers posting during a trip. One approach to counter these issues is to set a minimum $P(c|w)$ threshold

for cities, and further perform randomised sampling from $\mathbf{c}'$. In this chapter, we follow Chang *et al.* (2012) in constructing the final $\mathbf{c}'$: first, all cities containing w are ranked by $P(c|w)$ in decreasing order, then $\mathbf{c}'$ is formed by adding cities according to rank, stopping when the sum of $P(c|w)$ exceeds a pre-defined threshold r. We choose $r = 0.1$ in our experiments, based on the findings of Chang *et al.* (2012).

4.4 Benchmarking Experiments on NA

In this section, we compare and discuss the proposed feature selection methods. In particular, we investigate whether using only LIWs for geolocation prediction is better than using the full set of features, under various configurations of models and location partitions in Section 4.4.2. The subsequent experiments in this section are exclusively based on the public NA dataset. We adopt the same user partitions for training, dev and test as was used in the original paper (Roller *et al.* 2012). We primarily use the city-based class representation in our experiments over NA, but additionally present results using the original k-d tree partitions learned by Roller *et al.* (2012) in Section 4.4.2, for direct comparability with their published results. For the distance-based evaluation measures (Acc@161 and Median), we calculate the user's location based on the centroid of their tweets, and, depending on the class representation used, represent the predicted location as either: (a) a city centre; or (b) the user-centroid for a given k-d tree cell. In the case of Acc for the city-based class representation, we map the centroid for each user to the nearest city centre $\leq$ 50km away, and use this as the basis of the Acc calculation. In the case that there is no city centre that satisfies this constraint,[18] we map the user to the NULL class, and will always misclassify the user.[19]

[18]This occurs for 1139 ($\approx$ 11.4%) of test users.

[19]As such, the upper bound Acc for the city-based representation is 0.886. Note also that the Acc for the k-d tree vs. city-based representation is not comparable, because of the different class structure and granularity.

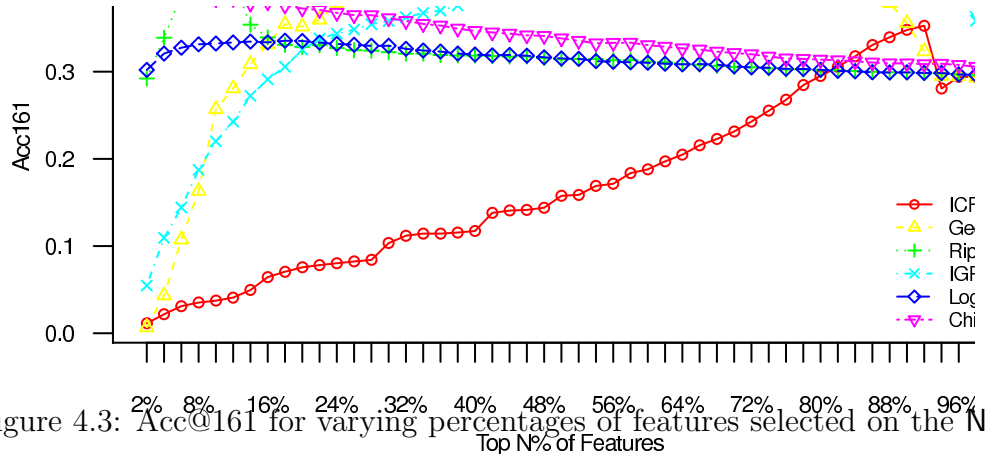


Figure 4.3: Acc@161 for varying percentages of features selected on the NA dataset, based on the city-based class representation.

4.4.1 Comparison of Feature Selection Methods

First, we compare the effectiveness of the various feature selection methods on NA using the city-based class representation. In total, 214K features were extracted from the training section of NA. We select the top-n% of features, with a step size of 2%, and then use the selected features within a multinomial naive Bayes learner (we return to explore the choice of learner in Section 4.4.2). The tuning of n for all methods is based on Acc@161 over the 10K held-out users in the development data. We present results for a sample of feature selection methods in Figure 4.3, omitting methods which are largely identical in behaviour to other methods presented in the graph, namely:

- $\{DistDiff_{tv}, DistDiff_{skl}\} \equiv ICF$
- $MaxChi \equiv Chi$
- $\{LR, IG, GeoSpread\} \equiv LogLike$

For all methods, the best result is achieved with a proper subset of features based on feature selection, although the proportion of the features that gives the best results for a given method varies greatly (e.g., the optima for *Ripley*, *IGR* and *GeoDen* are 10%, 88% and 66%, respectively). This observation agrees with the expectations that: (1) when only a small number of features is used, the trained model generally underfits the data; and (2) if the model is trained using the full feature set, noisy words (e.g., *the*) cause overfitting. For instance, when using just the top 2% of features in *IGR*, the most likely class for users with features — noting that users with no feature representation will default to the majority class, namely Los Angeles, US-CA — is Monterrey, MX, because Spanish words are highly location-indicative of the small number of Mexican cities in the NA dataset. The features which are selected last are generally high-frequency function words (e.g., *the*) and common words (e.g., *facebook*), which give little indication as to geolocation, and lead to prediction errors.

Two patterns can be observed in the results: (1) *Chi*, *MaxChi*, *IG*, *LogLike*, *GeoSpread*, *LR* and *Ripley* (i.e., "local" methods, which initially select features for each class, with the exception of *IG* and *Ripley*) achieve their highest Acc@161 at an early stage, then the numbers drop gradually; and (2) *ICF*, *IGR*, $DistDiff_{skl}$, $DistDiff_{tv}$ and *GeoDen* (i.e., the "collective" group, which select features for all classes at once) gradually increase in accuracy as more features are added, reach a peak when the majority of features are selected, then drop off in accuracy sharply. This difference in behaviour can be attributed to the types of word that are preferred by the methods. The "local" methods tend to prefer 1-local words — taking *LR*, for example, city names (e.g., *philadelphia*) and names of upper-level administrative regions (e.g., *georgia*) frequently occur in the upper reaches of the ranking. In addition to these gazetted words, many local/regional words are also found in the upper reaches of the feature ranking, including informal place names (e.g., *philly*, an informal name for Philadelphia, US-PA), local transport references (e.g., *skytrain*, a public transport system in Vancouver, CA) and local greetings (e.g., *aloha* in Honolulu, US-HI). However, it is reasonable to believe that 1-local words — words that are predominantly used in one city and are rarely mentioned in other cities — are not common. As a result, the accuracy is bounded by the limited number of true 1-local words. This

could be the reason for the early, yet remarkably high, peak in accuracy, and subsequent sharp decline, for *Ripley*; because of its reliance on pairwise distances between users using a given word, *Ripley* tends to rank 1-local words highly. In contrast, the "collective" methods assume words carry varying amounts of geospatial information. By leveraging combinations of LIWs, the true location of a user can be collectively inferred. For instance, *brunswick* is a common suburb/street name in many cities, e.g., Melbourne, AU and London, GB. This word alone is insufficient to make reliable predictions. However, if other LIWs (e.g., *tram* and *Flinders*, which are again not uniquely disambiguating in themselves) are also observed, then the chance of the location being Melbourne, AU becomes high, since it is unlikely that users from cities other than Melbourne, AU would use that combination of words. This strategy can also be explained in information-theoretic terms: by knowing more words, extra information is obtained, and consequently the entropy is continuously reduced and the prediction of geolocation becomes more certain.

Among all the feature selection methods, *IGR*, *GeoDen* and *Ripley* are the standout methods in terms of Acc@161. We further compare the accuracy of classifiers trained using the optimised set of LIWs (based on the development data) to that of the full model. The performance is measured on the 10K held-out test users, using the city-based class representation. The results are displayed in Table 4.3 (for the same subset of feature selection methods as were displayed in Figure 4.3), and show that using LIWs offers an improvement over the full feature set for all evaluation measures and all feature selection methods, except for slight dips in Acc@C for *IGR* and *GeoDen*. Nevertheless, these numbers clearly demonstrate that feature selection can improve text-based geolocation prediction accuracy. *IGR* performs best in terms of accuracy, achieving 8.9% and 14.2% absolute improvements in Acc and Acc@161, respectively, over the full feature set.

4.4.2 Comparison with Benchmarks

We further compare the best-performing method from Section 4.4.1 with a number of benchmarks and baselines. We experiment with two class representations: (1)

Dataset	Features	Acc	Acc@161	Acc@C	Median
	Full	0.171	0.308	0.831	571
	ICF	0.209	0.359	0.840	533
	Chi	0.233	0.402	**0.850**	385
NA	*IGR*	**0.260**	**0.450**	0.811	**260**
	LogLike	0.191	0.343	0.836	489
	GeoDen	0.258	0.445	0.791	282
	Ripley	0.236	0.432	0.849	306

Table 4.3: Results on the full feature set compared to that for each of a representative sample of feature selection methodologies on NA using NB with the city-based class representation. The best numbers are shown in boldface.

the city-based class representation based on `Geonames`; and (2) the k-d tree based partitioning of Roller *et al.* (2012), which creates grid cells containing roughly even amounts of data of differing geographical sizes, such that higher-population areas are represented with finer-grained grids.[20] For both class representations, we compare learners with and without feature selection. As observed previously, Acc is not comparable across the two class representations. Results based on the distance-based measures (Acc@161 and Median), on the other hand, are directly comparable. Acc@C results are not presented for the k-d tree based class representation because the k-d tree cells do not map cleanly onto national borders; although we could certainly take the country in which the centroid of a given k-d tree cell lies as the country label for the entire cell, such an approach would ignore known geo-political boundaries.

We consider the following methods:

Baseline: Because the geographical distribution of tweets is skewed towards higher-population areas (as indicated in Figure 4.1), we consider a most-frequent class baseline. We assign all users to the coordinates of the most-common city centre (or k-d tree grid centroid) in the training data.

[20]Recent work (Schulz *et al.* 2013) also considers irregular-sized polygons, based on administrative regions like cities.

Placemaker: Following Kinsella *et al.* (2011), we obtain results from Yahoo! Placemaker,[21] a publicly-available geolocation service. The first 50K bytes (the maximum query length allowed by Placemaker) from the tweets for each user are passed to Placemaker as queries. The returned city centre predictions are mapped to our collapsed city representations. For queries without results, or with a predicted location outside North America, we back off to the most-frequent class baseline.[22]

Multinomial naive Bayes: This is the same model as was used in Section 4.4.1.

KL divergence: The previous best results over NA were achieved using KL divergence and a k-d tree grid (Roller *et al.* 2012). Using a k-d tree, the earth's surface is partitioned into near-rectangular polygons which vary in size, but contain approximately the same number of users. Locations are represented as cells in this grid. KL divergence is then utilised to measure the similarity between the distribution of words in a user's aggregated tweets and that in each grid cell, with the predicted location being the centroid of the most-similar grid cell.[23]

Logistic regression: We also apply logistic regression from Section 4.3.4 as a learner. Instead of modelling all the data, we use only the *IGR*-selected features from Section 4.4.1. We experimented with both unregularised and L2-regularised logistic regression learners and found the results to be almost identical.[24] Based on this result and the fact that finding an appropriate regularisation function is non-trivial, we made a conscious choice not to use regularisation. Furthermore, the implementation of the regulariser would differ across learners and

[21] `http://developer.yahoo.com/geo/placemaker/`, (Retrieved 08/2012)

[22] An alternative would be to query Placemaker with each tweet, and then aggregate these predictions (e.g., by selecting the majority location) to get a final user-level prediction. However, Kinsella *et al.* (2011) found the accuracy of such an approach to be largely similar to that of the approach we use.

[23] We use the same settings as Roller *et al.* (2012): a median-based k-d tree partition, with each partition containing approximately 1050 users.

[24] The L2 regularisation is achieved by setting the Gaussian prior to 0.1, 1.0, 3.0, 5.0, 7.0, 9.0, 10.0, and 100.0.

Partition	Method	Acc	Acc@161	Acc@C	Median
City	Baseline	0.003	0.062	**0.947**	3089
	Placemaker	0.049	0.150	0.525	1857
	NB	0.171	0.308	0.831	571
	NB+*IGR*	**0.260**	**0.450**	0.811	**260**
	LR	0.129	0.232	0.756	878
	LR+*IGR*	0.229	0.406	0.842	369

Table 4.4: Geolocation performance using city-based partition on NA. Results using the optimised feature set (+*IGR*) are also shown. The best-performing method for each evaluation measure and class representation is shown in boldface.

Partition	Method	Acc	Acc@161	Acc@C	Median
k-d tree	Baseline	0.003	0.118	–	1189
	NB	0.122	0.367	–	404
	NB+*IGR*	0.153	0.432	–	280
	KL	0.117	0.344	–	469
	KL+*IGR*	**0.161**	**0.437**	–	**273**

Table 4.5: Geolocation performance using k-d tree-based partition on NA. Results using the optimised feature set (+*IGR*) are also shown. The best-performing method for each evaluation measure and class representation is shown in boldface.

complicate the direct comparison of feature selection methods (i.e., it would be difficult to tease apart the impact of the specific regulariser from the feature selection). Having said that, if the objective were to maximise the raw classifier accuracy — as distinct from exploring the impact of different features and feature selection methods on classification accuracy — we would advocate an extensive evaluation of different regularisers.

Instead of evaluating every possible combination of model, partition and feature set, we choose representative combinations to test the extent to which LIWs improve accuracy. The results on the city-based partition are shown in Table 4.4. We begin

by considering the baseline results. The most-frequent class for the city-based representation is Los Angeles, US-CA.[25] Both the majority class baseline and Placemaker perform well below multinomial naive Bayes (NB) and logistic regression (LR), and have very high Median distances. Furthermore, when using the features selected in Section 4.4.1 (i.e., NB+*IGR* and LR+*IGR*), the performance is further improved by a large margin for both models, demonstrating that identification of LIWs can improve text-based geolocation prediction. Finally, although LR performs poorly compared to NB, LR+*IGR* still improves substantially over LR. We plan to further explore the reasons for LR's poor performance in future work. Overall, NB+*IGR* performs best for the city-based representation in terms of Acc, Acc@161, and Median distance.

Turning to the k-d tree-based partition in Table 4.5, we again observe the low performance of the most-frequent class baseline (i.e., a grid cell near New York state). NB and KL — representative generative and discriminative models, respectively — are evaluated using software provided by Roller *et al.* (2012).[26] Both approaches clearly outperform the baseline over the k-d tree class representation. Furthermore, performance increases again when using the resultant feature set of LIWs,[27] demonstrating that for a variety of approaches, identification of LIWs can improve text-based geolocation.

Overall, compared to the previous published results for the k-d tree based representation (KL), *IGR*-based feature selection on the city-based partition achieves a 10.6% absolute improvement in terms of Acc@161, and reduces the Median prediction error by 209km.

From the results on the k-d tree based representation, it is not clear which of KL or NB is better for our task: in terms of Acc@161, NB outperforms KL, but KL+*IGR* outperforms NB+*IGR*. All differences are small, however, suggesting that the two methods are largely indistinguishable for the user geolocation task. As to the

[25]New York is further divided into suburbs, such as manhattan-ny061-us, brooklyn-ny047-us, in `Geonames`. As an artefact of this, these suburbs are not merged into a single city.

[26]`https://github.com/utcompling/textgrounder/wiki/RollerEtAl_EMNLP2012`

[27]Note that after LIWs are selected, a small proportion of users end up with no features. These users are not geolocatable in the case of KL, a discriminative model. We turn off feature selection for such users, and backoff to the full feature set, so that the number of test instances is consistent in all rows.

question of which class representation should be used for user geolocation, empirically, there seems to be little to separate the two, although further experimentation may shed more light on this issue. The city-based approach is intuitive, and enables a convenient country-level mapping for coarser-grained geolocation tasks. Furthermore, our observation from Figure 4.1 suggests most Twitter users are from cities. We therefore use the city-based partition for the remainder of this chapter for consistency and ease of interpretation.

A spin-off benefit of feature selection is that it leads to more compact models, which are more efficient in terms of computational processing and memory. Comparing the model based on LIWs selected using *IGR* with the full model, we find that the prediction time is faster by a factor of roughly five.

4.5 Experiments on WORLD

In addition to establishing comparisons on NA, we further evaluate the feature selection methods on WORLD. This extends the evaluation from regional benchmarks to global geolocation performance. Similar to NA, for WORLD we reserve 10K random users for each of dev and test, and the remainder of the users are used for training (preprocessed as described in Section 4.2.4). Here and in all experiments over WORLD and related datasets, we base our evaluation on the city label set.

We apply the same tuning procedure as was used over NA to obtain the optimal feature set for each feature selection method. We present results for a representative sample of the best-performing methods in Figure 4.4. Once again, we omit methods that are largely identical in behaviour to other methods, namely:

- $\{DistDiff_{tv}, DistDiff_{skl}\} \equiv ICF$
- $\{MaxChi, Chi, LogLike, IG, GeoSpread\} \equiv LR$

The biggest differences over Figure 4.3 are: (1) the χ^2-based methods converge in behaviour with *LR*, *LogLike* and related methods; and (2) *LR* performs marginally better than *LogLike*, and is thus the method we present in the graph.

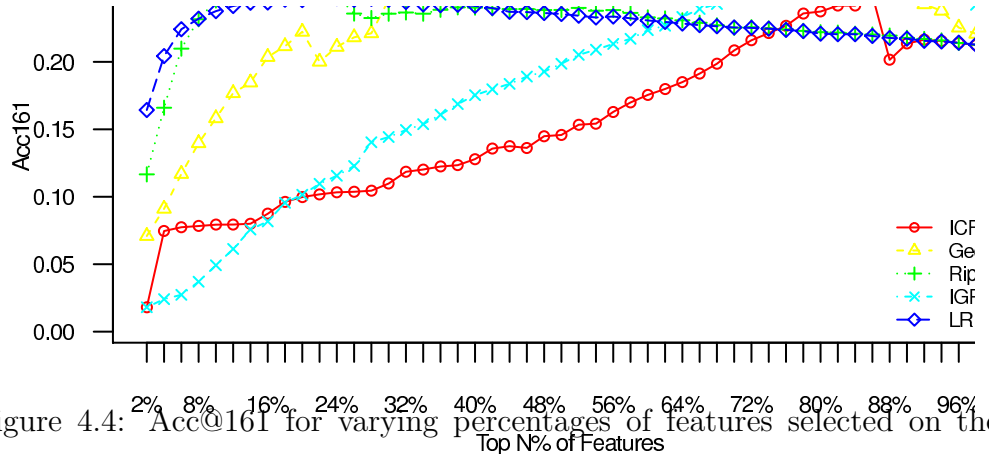


Figure 4.4: Acc@161 for varying percentages of features selected on the WORLD dataset, based on the city-based class representation.

Despite the difference in scope and data size, the overall trend over WORLD mirrors that for NA. In particular, *GeoDen*, *IGR* and *Ripley* achieve the best Acc@161 numbers on the dev data, although the numbers are lower than those achieved for NA in Figure 4.3. This is because WORLD has fewer tweets per user than NA (as we only utilise geo-tagged data), and disambiguation at the global level also makes it a more challenging task.

The results for multinomial naive Bayes with the chosen feature selection methods on WORLD are shown in Table 4.6. Again *GeoDen* (62%), *IGR* (86%) and *Ripley* (20%) achieve the best accuracy, although there is no clear winner: *IGR* achieves the best Acc and *Ripley* achieves the best Acc@161. Nevertheless, the improved city-based Acc and Acc@161 numbers confirm the general effectiveness of feature selection. On the basis of these similar results and the earlier NA results (in which *IGR* delivers better results), we adopt *IGR* as our default LIW feature selection method for the remainder of the chapter.

In summary, the findings on the utility of feature selection in Table 4.3 (NA) and

Dataset	Features	Acc	Acc@161	Acc@C	Median
WORLD	Full	0.081	0.200	**0.807**	886
	ICF	0.110	0.241	0.788	837
	IGR	**0.126**	0.262	0.684	913
	LR	0.104	0.233	0.792	**640**
	GeoDen	0.123	0.266	0.691	842
	Ripley	0.121	**0.268**	0.582	1128

Table 4.6: Results on the full feature set compared to that of each of a representative sample of feature selection methodologies on WORLD using NB with the city-based class representation. The best numbers are shown in boldface.

Table 4.6 (WORLD) tell a similar story, namely that feature selection improves user geolocation accuracy. The impact of feature selection on NA is much greater than WORLD, because WORLD has a larger number of classes and smaller average number of tweets per user and also per class, making it a more challenging dataset.

4.6 Exploiting Non-geotagged Tweets

Most Twitter-based geolocation research carried out to date (Eisenstein *et al.* 2010; Wing and Baldridge 2011) has been trained only on geotagged tweets, that is tweets with known geographical coordinates. Some work (Roller *et al.* 2012) has also incorporated non-geotagged tweets from users whose location can be inferred from geotagged tweets. Clearly, if it is possible to effectively utilise non-geotagged tweets, data sparsity can be ameliorated (as we aren't restricting ourselves to training on only the approximately 1% of tweets with known location), but there is a clear tradeoff in the confidence we can place in the labels associated with those tweets/users. In this section, we investigate the utility of non-geotagged tweets in geolocation prediction.[28]

For experiments in this section, and the rest of the chapter, we use WORLD+NG to

[28]Essentially, it is up to user whether to include/exclude GPS information, however, we don't have further information on how this decision is presented to the user on different platforms.

Train	Test	Acc	Acc@161	Acc@C	Median
G	G	0.126	0.262	0.684	913
G+NG	G	0.170	0.323	0.733	615
G	G+NG	0.187	0.366	0.835	398
G+NG	G+NG	0.280	0.492	0.878	170
G	NG	0.161	0.331	0.790	516
G+NG	NG	0.241	0.440	0.826	272
G	G-small	0.121	0.258	0.675	960
G	NG-small	0.114	0.248	0.666	1057

Table 4.7: Results of geolocation models trained and tested on geotagged (G) and non-geotagged (NG) tweets, and their combination.

denote the dataset which incorporates both the geotagged and non-geotagged tweets from the users in WORLD. We refer to the subparts of this dataset consisting of geotagged and non-geotagged tweets as G and NG, respectively. Of the 194M tweets in WORLD+NG, 12M are geotagged and the remaining 182M are non-geotagged. We use the same partitioning of users into training, development, and testing sets for WORLD+NG as for WORLD. We compare the relative impact of NG in which we train and test the geolocation method on G, NG, or their combination. Results are presented in Table 4.7.

The first row of Table 4.7 shows the results using only geotagged data (our best result from Table 4.6). In rows two and three, we show results when the data for each user in the training and test datasets, respectively, is expanded to incorporate non-geotagged data (without changing the set of users or the label for any user in either case). In both cases, for all evaluation measures, the performance is substantially better than the benchmark (i.e., the first row). This finding is in line with Cheng *et al.*'s (2010) results that data spareness is a big issue for text-based geolocation. It also validates our hypothesis that non-geotagged tweets are indicative of location. The best results are achieved when non-geotagged tweets are incorporated in both

the training and testing data (shown in row four). In this case we achieve an accuracy of 28.0%, a 15.4 percentage point increase over the benchmark using only geotagged tweets to represent a given user. Moreover, our prediction is within 161km of the correct location for almost one in every two users, and the country-level accuracy reaches almost 88%.[29]

Although research on text-based geolocation has used geotagged data for evaluation, the ultimate goal of this line of research is to be able to reliably predict the locations of users for whom the location is not known, i.e., where there is only non-geotagged data. Because geotagged tweets are typically sent via GPS-enabled devices such as smartphones, while non-geotagged tweets are sent from a wider range of devices, there could be systematic differences in the content of geotagged and non-geotagged tweets. We examine this issue in rows five and six of Table 4.7, where we test our model on only non-geotagged data. In this case we know a test user's gold-standard location based on their geotagged tweets. However these geotagged tweets are not used to represent the user in the test instance; instead, the user is represented only by their non-geotagged tweets. The results here are actually better than for experiments with the same training data but tested on geotagged tweets (i.e., rows one and two of the table).[30] This confirms that a model trained on G or G+NG indeed generalises to NG data. However, it is not clear whether this finding is due to there being much more non-geotagged than geotagged data for a given user, or whether some property of the non-geotagged data makes it easier to classify. To explore this question, we carry out the following additional experiment. First, we construct a new dataset NG-small by down-sampling NG to contain the same number of features per user as G (in terms of the feature token count). To make the comparison fairer, a second new dataset —G-small — is constructed, in which we exclude test users with more G tweets than NG tweets. This guarantees that users in NG-small will contain the same number of LIWs as in G-small. We average over five iterations of random

[29]Note that this evaluation is over exactly the same set of users in all four cases; all that changes is whether we incorporate extra *tweets* for the pre-existing set of users, in the training or test data.

[30]We remove users who only have geotagged tweets in the test data, reducing the number of users marginally from 10,000 to 9,767.

subsampling, and list the result in the final row of Table 4.7.[31] Here we see that the results for NG-small are not as good as G-small (i.e., row seven), suggesting that there might be minor sub-domain differences between geotagged and non-geotagged tweets, though a strong conclusion cannot be drawn without further in-depth analysis. One possible explanation is that there could be differences (e.g., demographic variations) between users who only have non-geotagged tweets and users who have both non-geotagged tweets and geotagged tweets; however, comparing these two sources is beyond the scope of this chapter. Nonetheless, the results suggest the difference between NG and G is largely due to the abundant data in NG. This explanation is also supported by the recent work of Priedhorsky *et al.* (2014).

In summary, we have quantitatively demonstrated the impact of non-geotagged tweets on geolocation prediction, and verified that models trained on geotagged data are indeed applicable to non-geotagged data, even though minor sub-domain differences appear to exist. We also established that representing a user by the combination of their geotagged and non-geotagged tweets produces the best results.

4.7 Language Influence on Geolocation Predication

Previous research on text-based geolocation has primarily focused on English data. Most studies have either explicitly excluded non-English data, or have been based on datasets consisting of primarily English messages, e.g., through selection of tweets from predominantly English-speaking regions (Eisenstein *et al.* 2010; Cheng *et al.* 2010; Wing and Baldridge 2011; Roller *et al.* 2012). However, Twitter is a multilingual medium and some languages might be powerful indicators of location: for example, if a user posts mostly Japanese tweets, this could be a strong indication that the user is based in Japan, which could be used to bias the class priors for the user. In this section, we explore the influence of language on geolocation prediction. The

[31] Note that we calculated the variance over the five iterations of random subsampling, and found it to be negligible for all evaluation measures.

predominant language in a given tweet was identified using `langid-2012`,[32] which has been trained to recognise 97 languages (Lui and Baldwin 2012).

To create a dataset consisting of multilingual geotagged tweets, we extract all geotagged data — regardless of language — from the same Twitter crawl that WORLD was based on. This multilingual dataset consists of 23M tweets from 2.1M users. 12M tweets are in English as in WORLD, while the remaining 11M tweets are in other languages. Figure 4.5 shows the proportion of tweets in the fifteen most common languages in the dataset.[33]

An immediate observation is the large difference in language distribution we observe for geo-tagged tweets as compared to what has been observed over all tweets (irrespective of geotag: (Hong *et al.* 2011; Baldwin *et al.* 2013)): among the higher-density languages on Twitter, there appears to be a weak positive bias towards English users geotagging their tweets, and a strong negative bias against Japanese, Korean and German users geotagging their tweets. We can only speculate that the negative bias is caused by stronger concerns/awareness of privacy issues in countries such as Japan, South Korea, Germany and Austria. We explored the question of whether this bias was influenced by the choice of Twitter client by looking at the distribution of Twitter clients used to post messages in each of English, German, Japanese and Korean: (a) overall (irrespective of whether the message is geotagged or not), based on a 1M sample of tweets from 28/09/2011; and (b) for geotagged tweets, based on WORLD. Overall, we found there to be huge variety in the choice of client used within a given language (with the top-10 clients accounting for only 65–78% of posts, depending on the language), and significant differences in popular clients between languages (e.g. "Keitai Web" is the most popular client for Japanese, "web" for English and German, and "Twitter for Android" for Korean). For geotagged tweets, on the other hand, there is much greater consistency, with the three most popular clients for all languages being "Twitter for iOS", "Twitter for Android" and "Foursquare", accounting for a relatively constant two-thirds of posts for each language. This is

[32]Based on the simplifying assumptions that: (a) every tweet contains linguistic content; and (b) all tweets are monolingual, or at least are predominantly in a single language.

[33]We represent languages in Figure 4.5 using two-letter ISO 639-1 codes.

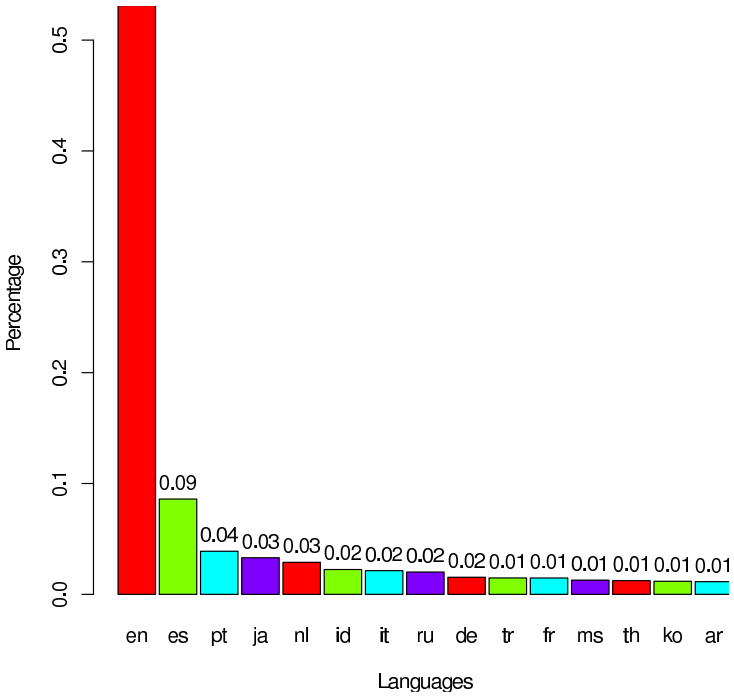


Figure 4.5: The percentage of tweets written in each of the fifteen most frequent languages. These fifteen languages account for 88% of the tweets in WORLD+ML.

suggestive of the fact that the choice of client is one factor in biasing the relative proportion of geotagged tweets in the different languages, although more research is required to fully understand this effect.

The training, development and test data is re-partitioned for the multilingual setting to stratify on language, and the resultant dataset is referred to as WORLD+ML. Again, the development and testing sets consist of 10K users each, with the remaining users in the training set as in WORLD. Although in Section 4.6 we showed that adding non-geotagged data improves geolocation accuracy, the experiments in this section are based only on geotagged data, because of the prohibitive computational cost of experimenting with a much larger dataset. Note that this doesn't limit the generalisability of our results, it simply means that we have to be careful to compare them to the monolingual results from Table 4.7 based on only geotagged tweets (the

first row).

We first compare geolocation performance in a multilingual setting with that in an English-only setting, a comparison that past work on geolocation has not considered. The data in WORLD+ML is further partitioned into two subsets — E and NE — according to whether the majority of a given user's tweets are in English or non-English, respectively. Of the 10K test users in WORLD+ML, 5,916 are English and 4,084 are non-English. One challenge with the multilingual setting of these experiments is tokenisation. Although rudimentary tokenisation of many languages such as English and French can be accomplished using whitespace and punctuation, tokenisation is much more challenging for languages such as Japanese and Chinese which do not represent word boundaries with whitespace. However, amongst the most-common languages on Twitter (as shown in Figure 4.5), Japanese is the only language which accounts for a substantial portion of the data ($> 1\%$) and requires a specialised tokenisation strategy (compared to English). For Japanese tweets we apply the Japanese morphological segmenter `MeCab` (with the IPA dictionary),[34] and post-correct tokenisation errors relating to Twitter-specific tokens such as mentions, hashtags, and URLs (e.g., in instances where `MeCab` over-segments a mention into multiple morphemes). For non-Japanese tweets, we apply the same tokeniser based on regular expressions used in our previous English-only experiments.[35]

After resolving the tokenisation issue, we apply the same *IGR* method from Section 4.3.4 to select the optimised feature selection cut-off, based on Acc over the development data. We observe that a much larger proportion of tokens are selected in the multilingual setting compared to the English-only experiments. For example, of the 400K token types in the multilingual experiment, 384K (the top 96%) are selected as location-indicative, while for the English-only case 83K (the top 86%) location-indicative words are selected from the total of 96K token types.

The experimental results are shown in Table 4.8.[36] The first row gives results

[34] `http://sourceforge.net/projects/mecab/`

[35] In addition to token-based n-grams, we could also try character-based n-grams. Character-based n-grams are useful to capture lexical variants, such as *melb* and *melbourne*, which have common low order character n-grams. We do apply character-based n-grams for user-declared locations (in Section 4.8.3), in which the text is shorter and contains many such variants.

[36] The English-only results reported here are not the same as for the comparable experiment in

Train	Test	Acc	Acc@161	Acc@C	Median
E+NE	E+NE	0.196	0.343	0.772	466
E+NE	E	0.134	0.256	0.715	1067
E+NE	NE	0.287	0.468	0.855	200
E	E	0.169	0.317	0.746	632

Table 4.8: Results for multilingual geolocation prediction, training and testing on English (E) and non-English (NE) users, and their combination.

for training and testing on the full dataset of both English and non-English tweets. The next two rows show the results when testing on English (E) and non-English (NE) subsets of the data. The much lower accuracy for E compared to NE indicates that English tweets are much more difficult to geolocate than non-English tweets. One reason for this is that for many non-English languages, there is a strong bias towards a small number of cities. We verify this by calculating the class entropy with respective to a language on the training data. The class probabilities are smoothed using a simple add-α method, with $\alpha = 1/3709$ (where 3709 is the size of the class set). As shown in Table 4.9, the class entropy on English (en) data is the largest, indicating that English is prevalent across a large number of locations. In contrast, Thai (th) and Turkish (tr) have much smaller entropies, suggesting the location distributions are heavily skewed, and user geolocation over these languages will be easier than for English.

To explore the extent to which the geolocatability of a user varies with respect to the predominant language of their tweets, we further break down the results by language in Table 4.10, which shows results for the top-10 most frequent languages (by number of tweets) with at least 100 users in our test data. This cut-off on users ensures we do not consider under-represented languages.

We observe that the results vary remarkably by language in the multilingual sec-

Table 4.7 using only geotagged data, because the test sets consist of different users in these two cases.

Language	Entropy	Language	Entropy	Language	Entropy
en	6.279	id	3.868	fr	5.538
es	5.069	it	5.244	ms	3.970
pt	4.144	ru	3.772	th	2.697
ja	3.523	de	6.207	ko	2.781
nl	3.820	tr	2.888	ar	3.281

Table 4.9: Geolocation class entropy for the top fifteen languages in WORLD+ML.

tion of Table 4.10. The results are overall lowest for English (en), although the lowest country-level accuracy is for Arabic (ar); we speculate that this is caused by the large number of countries that Arabic is spoken in, and the relatively small number of Arabic speakers in our training data. Furthermore, the city-level accuracy is better than 30% for Indonesian (id), Japanese (ja), Russian (ru), Turkish (tr) and Arabic (ar); the regions in which these languages are commonly-spoken are more geographically-restricted than for English, suggesting that geolocation accuracy on languages with smaller geographic footprints will tend to be higher than for languages which are widely-used throughout a larger geographical area. This finding agrees with the recent work of Priedhorsky *et al.* (2014), and further indicates the power of language information in predicting locations. The best city-level accuracy of 53.8% is observed for Turkish (one of the languages with the lowest city-level entropy). Manually inspecting the outputs, we find that this is because our model predicts the city Istanbul for all Turkish users, and a large proportion of Turkish tweets come from this city.

Based on this finding, we further consider a language-based benchmark which predicts the most frequent city given the predominant language of a user's tweets (denoted as Per-language Majority Class). We also observe the performance gap between the multilingual model on English (the second row of Table 4.8) and an English-only model (the bottom row in Table 4.8). These results show that if the target data is known to be written in a single language then a monolingual model outperforms a multilingual one. It also suggests an alternative approach for multilin-

		Per-language Majority Class				Unified Multilingual				Monolingual Partitioning			
Lang.	No.	Acc	Acc@161	Acc@C	Med.	Acc	Acc@161	Acc@C	Med.	Acc	Acc@161	Acc@C	Med.
en	5916	0.019	0.039	0.655	3671	0.134	0.256	0.715	1067	0.169	0.317	0.746	632
es	945	0.116	0.159	0.324	4267	0.267	0.346	0.734	391	0.362	0.478	0.802	185
pt	673	0.223	0.296	0.952	490	0.232	0.305	0.952	490	0.400	0.489	0.961	200
id	398	0.264	0.472	0.899	197	0.324	0.565	0.965	115	0.440	0.736	0.960	16
nl	342	0.175	0.789	0.889	87	0.173	0.789	0.889	87	0.298	0.871	0.845	58
ja	298	0.326	0.530	0.960	96	0.336	0.544	0.956	95	0.463	0.695	0.950	27
ru	217	0.336	0.378	0.857	633	0.346	0.387	0.862	633	0.341	0.378	0.862	633
tr	186	0.538	0.656	0.930	0	0.538	0.656	0.930	0	0.522	0.645	0.930	0
ar	164	0.335	0.470	0.463	379	0.354	0.488	0.500	301	0.457	0.591	0.750	21
th	154	0.325	0.766	0.981	20	0.279	0.623	0.792	41	0.325	0.766	0.974	30
All	10000	0.107	0.189	0.693	2805	0.196	0.343	0.772	466	0.255	0.425	0.802	302

Table 4.10: Geolocation performance and comparison for the top ten most frequent languages in the multilingual test data, using (1) language prior (i.e., the city where a language is mostly used); (2) a unified multilingual model (i.e., training and testing on multilingual data regardless of languages); and (3) language-partitioned monolingual models (i.e., first identify the primary language of users, train one model per language, and classify test users with the model corresponding to the language of their tweets).

gual geolocation prediction: rather than training and predicting on multilingual data (E+NE), we can train and evaluate models on language-specific data. Motivated by this observation, we also apply a monolingual partitioned model for users of a particular language based on `langid-2012` (i.e., language partitions), e.g., selecting all Japanese users in the training data, and only applying the Japanese-specific model to Japanese users in the test data. This is denoted as Monolingual Partitioning in Table 4.10, and is contrasted with the simple approach of a combined model for all languages and users ("Unified Multilingual").

By comparing the Per-language Majority Class with the Unified Multilingual model, we find that the unified model performs better overall, with the exception of Thai (th) and Dutch (nl), both of which are associated with a very small number of cities, and one city which is much larger than the others (Bangkok, TH and Amsterdam, NL, respectively). Because of the relatively poor results for this benchmark method on languages such as English (en) and Spanish (es) which are frequent on Twitter, and its relatively poor overall performance, the Per-language Majority Class is not an appropriate method for this task. Nevertheless, when using a Monolingual Partitioning model, the results are far superior, and the partitioning effect of language can be seen. This suggests that modelling each language independently can improve geolocation performance.

In summary, this series of experiments has shown the influence of language on geolocation prediction. Among the top-10 languages found on Twitter, English is the most difficult to perform user geolocation over, as English is the most global language. Despite language variance, multilingual geolocation prediction is certainly feasible, although the best way to leverage language for geolocation prediction is by training language-partitioned monolingual models and geolocating users based on their primary language.

4.8 Incorporating User Meta Data

The metadata accompanying tweets is a valuable source of geographical information beyond that available in tweets. In this section, we explore incorporating

metadata information into our text-based geolocation system. We begin by selecting four metadata fields that could potentially provide insights into the location of a user, and first evaluate models trained on each of these sources of information. We then consider a number of ways to incorporate information from this metadata with our best text-based method developed in Section 4.6. As discussed in Section 4.7, language has a strong influence on geolocation prediction, and English-posting users are the hardest to geolocate. As such, we experiment only on English data (i.e., WORLD+NG) for the remainder of this chapter.

4.8.1 Unlocking the Potential of User-declared Metadata

We choose the following four user-supplied metadata fields for our study: location (LOC), timezone (TZ), description (DESC), and the user's real name (RNAME).[37] In contrast to rich social network information which is much more expensive to extract, these metadata fields are included in the JSON object that is provided by the Twitter Streaming API, i.e., we can extract this metadata at no extra crawling cost. This information, however, is dynamic, i.e., users can change their profiles, including the metadata of interest to us. By aggregating the extracted tweet-level metadata for each user, we can calculate the ratio of users that change each metadata field. 18% of users changed their DESC field during the approximately five months over which our dataset was collected. During this same time period, for each of the other fields considered, less than 8% of users updated their data. Given the relatively small number of profile updates, we ignore the influence of these changes, and use the most frequent value for each metadata field for each user in our experiments.

All of this user-supplied metadata can be imprecise or inaccurate, because the user is free to enter whatever textual information they choose. For example, some LOC fields are not accurate descriptions of geographical locations (e.g., *The best place in the universe*). Moreover, although some LOC fields are canonical renderings of a user's true location (e.g., *Boston, MA, USA*), a large number of abbreviations and

[37]The user-supplied real name could be any name — i.e., it is not necessarily the user's actual name — but is a different field from the user's screen name.

Data	LOC	TZ	DESC
Training	0.813	0.752	0.760
Test	0.813	0.753	0.761

Table 4.11: The proportion of users with non-empty metadata in WORLD+NG.

non-standard forms are also observed (e.g., *MEL* for Melbourne, AU). Cheng *et al.* (2010) find that only a small proportion of location fields in their US-based dataset are canonical locations (i.e., of the form *city, state*). Nevertheless, these non-standard and inaccurate location fields might still carry information about location (Kinsella *et al.* 2011), similarly to how the text of tweets can indicate location without explicitly mentioning place names.

These metadata fields also differ with respect to the explicitness of the location information they encode. For instance, while LOC and TZ can give direct location information, DESC might contain references to location, e.g., *A geek and a Lisp developer in Bangalore*. Although RNAME does not directly encode location there are regional preferences for names (Bergsma *et al.* 2013), e.g., *Petrov* might be more common in Russia, and the name *Hasegawa* might be more common in Japan. Finally, for all of the tweets that we consider, the text field (i.e., the content of the tweet itself) and RNAME are always present, but LOC, TZ, and DESC can be missing if a user has chosen to not supply this information. The proportion of non-empty metadata fields for LOC, TZ and DESC for users in WORLD+NG are listed in Table 4.11.

4.8.2 Results of Metadata-based Classifiers

Because of the variable reliability and explicitness of the selected metadata, we incorporate these fields into our statistical geolocation model in a similar manner to the message text. In preliminary experiments, we considered bag-of-words features

Classifier	Acc	Acc@161	Acc@C	Median
LOC	0.405	0.525	0.834	92
TZ	0.064	0.171	0.565	1330
DESC	0.048	0.117	0.526	2907
RNAME	0.045	0.109	0.550	2611
BASELINE	0.008	0.019	0.600	3719
TEXT	0.280	0.492	0.878	170

Table 4.12: The performance of NB classifiers based on individual metadata fields, as well as a baseline, and the text-only classifier with *IGR* feature selection.

for the metadata fields, as well as bag-of-character n-gram features for $n \in \{1, ..., 4\}$.[38] We found character 4-grams to perform best, and report results using these features here. (A bag-of-character 4-grams represents the frequency of each four-character sequence including a start and end symbol.) The geolocation performance of a classifier trained on features from each metadata field in isolation, as well as the performance of a most frequent city baseline (BASELINE) and our best purely text-based classifier (TEXT, replicated from Table 4.7), is shown in Table 4.12.

The classifier based on each metadata field outperforms the baseline in terms of Acc, Acc@161, and Median error distance. This suggests these metadata fields do indeed encode geographically-identifying information, though some classifiers are less competitive than TEXT. Notably, despite the potential for noise in the user-supplied location fields, this classifier (LOC) achieves even better performance than the purely text-based method, reaching a city-level accuracy of over 40%, predicting a location within 161km of the true location for over half of the users. This suggests LOC contains valuable information, even though LOC fields are noisy (Cheng *et al.* 2010), and are not easily captured by off-the-shelf geolocation tools (Hecht *et al.* 2011).

[38]Although we could certainly also consider character n-grams for the text-based classifier, we opted for a bag-of-words representation because it explicitly captures the LIWs that we believe are especially important for geolocation. There could also be location-indicative character n-grams, the exploration of which we leave for future work.

TEXT	0.461	0.689	0.702	0.704
0.181	LOC	0.577	0.578	0.581
0.066	0.063	TZ	0.903	0.907
0.067	0.041	0.085	DESC	0.923
0.065	0.049	0.080	0.088	RNAME

Table 4.13: Pairwise correlation of base classifiers using Cohen's Kappa (bottom left) and Double Fault Measure (top right).

Manual analysis suggests many vernacular place names are captured in the statistical modelling, such as *Kiladelphia* and *Philly* used to represent *Philadelphia*. The utility of metadata fields is also confirmed by the recent work of Priedhorsky *et al.* (2014).

4.8.3 Ensemble Learning on Text-based Classifiers

To further analyse the behaviour of the four metadata classifiers, we consider the pairwise city-level prediction agreement between them. Cohen's Kappa is a conventional metric to evaluate inter-annotator agreement for categorical items (such as the predicted cities in our case); larger Kappa values indicate higher pairwise agreement. The double fault measure (Zhou 2012) incorporates gold-standard information, and is equal to the proportion of test cases for which both classifiers make a false prediction. This measure offers the empirical lowest error bound for the pairwise ensemble classifier performance.

Pairwise scores for Cohen's Kappa and the double fault measure are shown in Table 4.13. The Kappa scores (bottom-left of Table 4.13) are very low, indicating that there is little agreement between the classifiers. Because the classifiers achieve better than baseline performance, but also give quite different outputs, it might be possible to combine the classifiers to achieve better performance. The double fault results (top-right) further suggest that improved accuracy could be obtained by combining

classifiers.

We combine the individual classifiers using meta-classification. We first adopt a feature concatenation strategy that incrementally combines the feature vectors of TEXT, LOC, TZ, DESC and RNAME. We also consider *stacked generalisation* (Wolpert 1992), referred to simply as *stacking*, in which the outputs from the base classifiers, and the true city-level locations, are used to train a second classifier which produces the final output. The base classifiers, and the second classifier, are referred to as the *L0* and *L1* classifiers, respectively. In conventional applications of stacking, homogeneous training data is used to train heterogeneous *L0* classifiers; in our case, however, we train homogeneous *L0* multinomial Bayes models on heterogeneous data (i.e., different types of data such as TEXT, LOC, and TZ). We consider logistic regression (Fan *et al.* 2008) and multinomial Bayes as the *L1* classifier.

We carry out 10-fold cross validation on the training users to obtain the *L1* (final) classifier results, a standard procedure for stacking experiments. We use stratified sampling when partitioning the data because the number of users in different cities varies remarkably, and a simple random sample could have a bias towards bigger cities. The ensemble learning results are tabulated in Table 4.14.[39]

The combination of TEXT and LOC is an improvement over LOC (i.e., our best results so far). However, using feature concatenation and multinomial naive Bayes stacking, accuracy generally drops as metadata feature sets that perform relatively poorly in isolation (i.e., TZ, DESC, RNAME) are incorporated. On the other hand, using logistic regression stacking, we see small increases in accuracy as features that perform less well in isolation are incorporated. Though DESC and RNAME are moderately useful (as shown in Table 4.12), these fields contribute little to the strong ensembles (i.e., TEXT, LOC and TZ). The best model (using logistic regression stacking and all features) assigns users to the correct city in almost 50% of the test cases, and has a Median error of just 9km. Moreover, with this approach the country-level accuracy reaches almost 92%, indicating the effectiveness of our method for this coarse-grained geolocation task.

[39]Note that we map users to city centres and this may cause low Median distance. If a user is predicted to the correct city, the prediction error distance would be zero.

Feature concatenation

	Features	Acc	Acc@161	Acc@C	Median
1.	TEXT +LOC	0.444	0.646	0.923	27
2.	1. + TZ	0.429	0.639	0.929	32
3.	2. + DESC	0.319	0.529	0.912	127
4.	3. + RNAME	0.294	0.503	0.912	156

Multinomial Bayes stacking

	Features	Acc	Acc@161	Acc@C	Median
1.	TEXT +LOC	0.470	0.660	0.933	19
2.	1. + TZ	0.460	0.653	0.930	23
3.	2. + DESC	0.451	0.645	0.931	27
4.	3. + RNAME	0.451	0.645	0.931	27

Logistic regression stacking

	Features	Acc	Acc@161	Acc@C	Median
1.	TEXT +LOC	0.483	0.653	0.903	14
2.	1. + TZ	0.490	0.665	0.917	9
3.	2. + DESC	0.490	0.666	0.919	9
4.	3. + RNAME	0.491	0.667	0.919	9

Table 4.14: The performance of classifiers combining information from text and metadata using feature concatenation (top), multinomial Bayes stacking (middle), and logistic regression stacking (bottom). Features such as "1. + TZ" refer to the features used in row "1." in combination with TZ.

It is interesting to observe that, while we found NB to outperform LR as a standalone classifier in Section 4.4.2, as an L1 classifier, LR clearly outperforms NB. The reason for this is almost certainly the fact that we use a much smaller feature set relative to the number of training instances in our stacking experiments, under which circumstances, discriminative models tend to outperform generative models (Ng and Jordan 2002).

4.9 Temporal Influence on Geolocation Model

In addition to the held-out English test data in WORLD+NG, we also developed a new geotagged test dataset to measure the impact of time on model generalisation. The training and test data in WORLD+NG are time-homogeneous as they are randomly partitioned based on data collected in the same period. In contrast, the new test dataset (LIVE) is much newer, collected more than 1 year later than WORLD+NG. Given that Twitter users and topics change rapidly, a key question is whether the statistical model learned from the "old" training data is still effective over the "new" test data? This question has implications for the maintenance and retraining of geolocation models over time. In the experiments in this section we train on WORLD+NG and test on our new dataset.

The LIVE data was collected over 48 hours from 03/03/2013 to 05/03/2013, based on geotagged tweets from users whose declared language was English. Recent status updates (up to 200) were crawled for each user, and `langid-2012` was applied to the data to remove any remnant non-English messages. In addition to filtering users with less than 10 geotagged tweets for the test data as in WORLD+NG, we further exclude users with less than 50% of geotagged tweets from one city. This is because if a user's geotagged tweets are spread across different locations, it is less credible to adopt the user's most frequent location as their true primary location in evaluation. A post-check on the WORLD+NG test data shows that 9,977 out of 10K users satisfy this requirement on geographical coherence, and that we aren't unnecessarily biasing the data in LIVE in applying this criterion. Finally, all status updates are aggregated at the user-level, as in WORLD+NG. After filtering, 32K users were obtained, forming the final LIVE dataset.

We use only TEXT, LOC and TZ in this section, as they require less computation and achieve accuracy comparable to our best results, as shown in Table 4.14. The temporal factor impact on geolocation prediction model generalisation is revealed in the accuracy for WORLD+NG and LIVE shown in Table 4.15. Acc and Acc@161 numbers in the stacked model (1. + 2. + 3.) drop by approximately 8 and 5 percentage points, respectively, on LIVE as compared to WORLD+NG. The Median prediction

WORLD+NG				
Features	Acc	Acc@161	Acc@C	Median
1. TEXT	0.280	0.492	0.878	170
2. LOC	0.405	0.525	0.834	92
3. TZ	0.064	0.171	0.565	1330
1. + 2. + 3.	0.490	0.665	0.917	9
LIVE				
Features	Acc	Acc@161	Acc@C	Median
1. TEXT	0.268	0.510	0.901	151
2. LOC	0.326	0.465	0.813	306
3. TZ	0.065	0.160	0.525	1529
1. + 2. + 3.	0.406	0.614	0.901	40

Table 4.15: Generalisation comparison between the time-homogeneous WORLD+NG and time-heterogeneous LIVE (1. + 2. + 3. denotes stacking over TEXT, LOC and TZ).

error distance also increases moderately from 9km to 40km. By decomposing the stacked models and evaluating against the base classifiers, we find the accuracy declines are primarily caused by accuracy drops in the LOC classifier on the new LIVE data, of approximately 9% in Acc and 6% in Acc@161. This could be viewed as a type of over-fitting, in that the stacked classifier is relying too heavily on the predictions from the LOC base classifier. The TZ classifier performs relatively constantly in terms of accuracy, although the Median error increases slightly. The TEXT classifier is remarkably robust, with all numbers except for Acc improving marginally.

We further investigate the poor LOC classifier generalisation on LIVE. First, we down-sample LIVE to 10K users, the same size as WORLD+NG, and then compare the per-city prediction numbers on the two datasets using only the LOC classifier. We find two factors jointly cause the accuracy decrease on LIVE: (1) the composition of test users, and (2) the decline in per-city recall. For instance, 80 test users are from London, GB in WORLD+NG. This number sharply increases to 155 in LIVE, meaning that the influence of London, GB test users on the overall accuracy in LIVE

is almost doubled. Furthermore, the recall — the proportion of users from a given location who are correctly predicted as being from that location — for London, GB drops from 0.676 in WORLD+NG to 0.568 in LIVE. We observe that the proportion of empty LOC fields among London, GB test users jumps from 13% (WORLD+NG) to 26% (LIVE). This reduces the utility of the LOC data in LIVE and explains why the per-city recall drops: all test users with an empty LOC field are assigned to the city with highest class prior in the model (i.e., Los Angeles, US). Overall, the ratios of empty LOC fields in WORLD+NG test data and LIVE are 0.176 and 0.305, respectively, suggesting that user-declared locations in LIVE carry much less geospatial information than in WORLD+NG. We show other comparisons for the top-10 cities in terms of test users in Table 4.16,[40] as the accuracy of more highly-represented cities has a greater impact on overall results than that of smaller cities. Like London, GB, most cities shown in Table 4.16 experience lower recall scores for LIVE, and many of them have more test users in LIVE than in WORLD+NG. Nevertheless, some cities have higher recall and more test users in LIVE, e.g., Los Angeles, US and Anaheim, US in Table 4.16. The overall numbers are, of course, determined by aggregated performance over all cities. To provide some insight, 35.6% of cities in WORLD+NG have more than 40% in recall, but the number is only 28.5% in LIVE.

As an important base classifier in the stacked model, the LOC accuracy numbers are most influenced by temporal change. There are two possible explanations for the performance decline: (1) the prior location distribution $P(c)$ changed as demonstrated in Table 4.16; and (2) the posterior distribution of a word given a city label $P(w|c)$ could have changed. Either way, a periodically retrained LOC classifier would, no doubt, go some way towards remedying the temporal gap. Overall, the numbers suggest that time-homogeneous data (WORLD+NG) is easier to classify than time-heterogeneous data (LIVE). However, training on "old" data and testing on "new" data has been shown to be empirically viable for the TEXT and TZ base classifiers in particular. This result also validates efforts to optimise text-based user geolocation

[40]We observe that the city proportions changed drastically between WORLD+NG and LIVE. The reasons for this are unclear, and we can only speculate that it is due to significant shifts in microblogging usage in different locations around the world.

		LIVE		WORLD+NG	
Rank	cities in LIVE	users	recall	users	recall
1	Los Angeles, US	201	0.766	81	0.691
2	Kuala Lumpur, MY	168	0.482	50	0.560
3	London, GB	155	0.568	80	0.675
4	Jakarta, ID	129	0.550	86	0.686
5	Anaheim, US	85	0.447	26	0.346
6	Singapore, SG	76	0.474	160	0.556
7	Fort Worth, US	76	0.289	35	0.371
8	Chicago, US	72	0.569	123	0.577
9	Pittsburgh, US	72	0.431	39	0.487
10	San Antonio, US	66	0.455	82	0.585

Table 4.16: The recall and number of test users, by city, for the top ten largest cities in LIVE, compared with WORLD+NG.

classification accuracy. Recently, similar results on tweet-level geolocation prediction were observed by Priedhorsky *et al.* (2014), supporting the claim that the accuracy of geolocation prediction suffers from diachronic mismatches between the training and test data.

4.10 User Tweeting Behaviour

Having improved and extended text-based geolocation prediction, we now shift our focus to user geolocatability. If a user wishes to keep their geolocation private, they can simply disable public access of their tweets and metadata. However, if users choose to share their (non-geotagged) tweets, are there different tweeting behaviours which will make them more susceptible to geolocation privacy attacks? To investigate this question, in this section, we discuss the impact of user behaviour on geolocation accuracy relative to predictions over LIVE based on the stacking model

from Section 4.9.[41]

As an obvious first rule of thumb, geotagged tweets should be avoided, because they provide immediate access to a user's geographical footprint, e.g., favourite bars, or their office address. Second, as an immediate implication of our finding that location metadata is a strong predictor of geolocation (Section 4.8.2), if a user wants to avoid privacy attacks, they should avoid presenting location metadata, in effect disabling the LOC base classifier in our stacked classifier. Third, the text of a user's posts can be used to geolocate the user (at approximately 27% Acc, from Table 4.15). To investigate the impact of the volume of tweets on user "geolocatability", we perform a breakdown of results over LIVE across two dimensions: (1) the number of LIWs, to investigate whether the sheer volume of tweets from a user makes them more geolocatable; and (2) the source of geospatial information which we exploit in the geolocation model. We evaluate these questions in Figure 4.6 in four feature combination settings, relative to the: (1) tweet text-based classifier; (2) tweet text-based classifier with gazetteer names removed;[42] (3) metadata stacking using LOC and TZ (invariant to tweet number changes); and (4) the stacking of TEXT, LOC and TZ for all users. In each case, we partition the data into 20 partitions of 5% of users each, ranked by the total number of LIWs contained in the combined posts from that user. In addition to the Acc for each user partition, we also indicate the average number of LIWs per user in each partition (as shown in the second y-axis, on the right side of the graph).

Overall, the more LIWs are contained in a user's tweets, the higher the Acc for text-based methods. When gazetted terms are removed from the tweets, Acc drops by a large margin. This suggests gazetted terms play a crucial role in user geolocation. Metadata also contributes substantially to accuracy, improving the text-based accuracy consistently. Moreover, if a user tweets a lot, the Acc of the tweet text-based approach is comparable to our best model, even without access to the metadata

[41] Our analysis is limited to behaviours that could easily be adopted by many users. Given that our system predicts the most likely city from a fixed set for a given user, one simple way to avoid being geolocated is to move far away from any of these cities. However, it seems unlikely that this strategy would be widely adopted.

[42] Our gazetteer is based on the ASCII city names in the `Geonames` data.

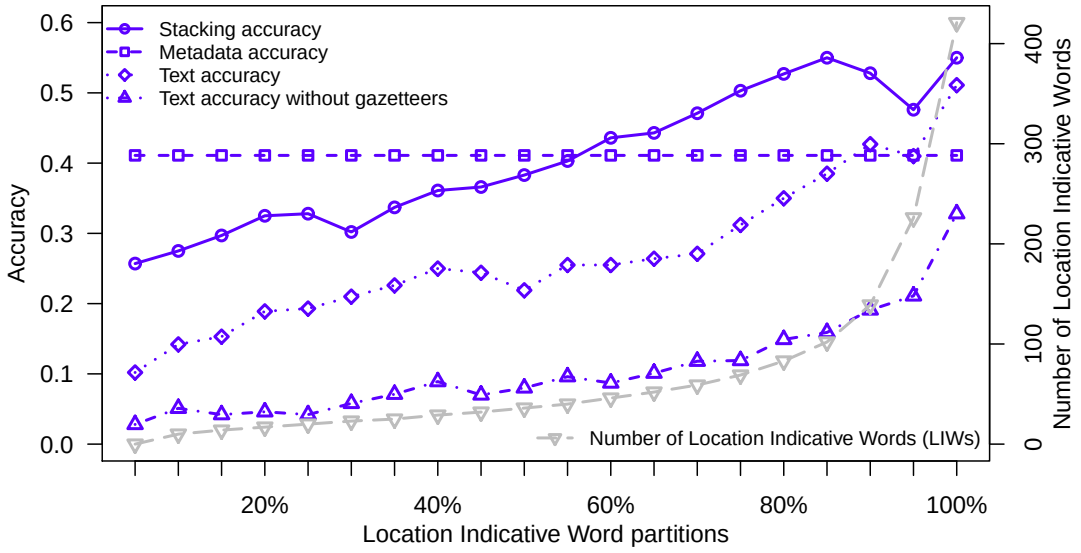


Figure 4.6: The impact of the use of LIWs on geolocation accuracy. Users are sorted by the number of LIWs in their tweets, and are partitioned into 20 bins. Metadata includes LOC and TZ.

(as shown in the top right corner of the graph). As an overall recommendation, users who wish to obfuscate their location should leave the metadata fields blank and avoid mentioning LIWs (e.g., gazetted terms and dialectal words) in their tweets. This will make it very difficult for our best geolocation models to infer their location correctly (as demonstrated to the bottom left of the graph). A similar conclusion on user geolocatability was recently obtained by Priedhorsky *et al.* (2014). To help privacy-conscious Twitter users to avoid being geolocated by their tweets, we have made the list of LIWs publicly available.[43]

[43] `http://www.csse.unimelb.edu.au/~tim/etc/liw-jair.tgz`

4.11 Prediction Confidence

In the task setup to date, we have forced our models to geolocate all users. In practice, however, many users don't explicitly mention any geolocating words in their posts, making the task nigh on impossible even for a human oracle. An alternative approach would be to predict a user geolocation only when the model is confident of its prediction. Here, we consider a range of variables that potentially indicate the prediction confidence.

Absolute probability (AP): Only consider predictions with probability above a specified threshold.

Prediction coherence (PC): We hypothesise that for reliable predictions, the top-ranked locations will tend to be geographically close. In this preliminary exploration of coherence, we formulate PC as the sum of the reciprocal ranks of the predictions corresponding to the second-level administrative region in our class representation (i.e., state or province) of the top-ranking prediction, calculated over the top-10 predictions.[44] For example, suppose the top-10 second-level predictions were in the following states in the US: US-TX, US-FL, US-TX, US-TX, US-CA, US-TX, US-TX, US-FL, US-CA, US-NY. The top-ranking state-level prediction is therefore US-TX, which also occurs at ranks 3, 4, 6 and 7 (for different cities in Texas). In this case, PC would be $\frac{1}{1} + \frac{1}{3} + \frac{1}{4} + \frac{1}{6} + \frac{1}{7}$.

Probability ratio (PR): If the model is confident in its prediction, the first prediction will tend to be much more probable than other predictions. We formulate this intuition as PR, the ratio of the probability of the first and second most-probable predictions.

Feature number (FN): We take the number of features found in a user's posts as the prediction accuracy. The intuition here is that a geolocation prediction based on more features is more reliable than a prediction based on fewer features.

[44] It could be measured by the average distance between top predictions as well.

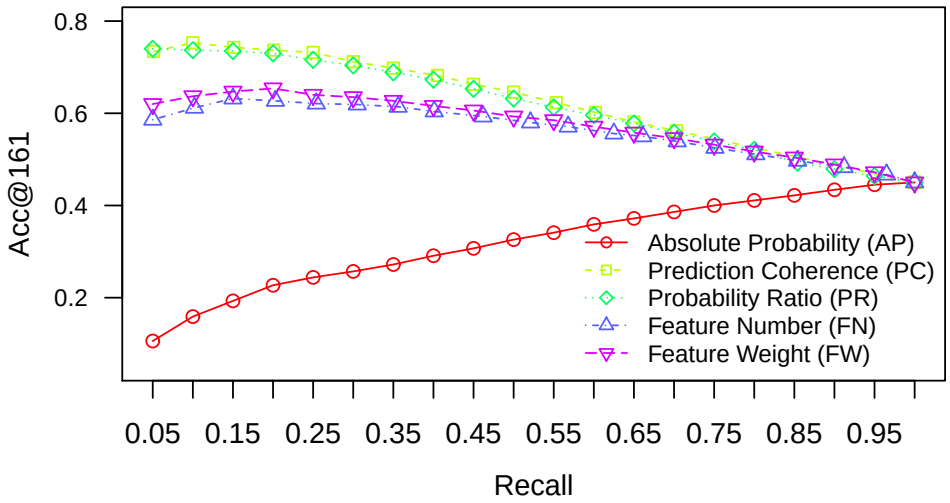


Figure 4.7: Acc@161 for classification of the top-n% most-confident predictions for each measure of text-based prediction confidence on NA.

Feature weight (FW): Similar to FN, but in this case we use the sum of *IGR* of all features, rather than just the number of features.

We investigate these variables on both NA and LIVE results. In particular, we only evaluate them using the text-based model, as we experiment only with text-based user geolocation in this section. Nevertheless, exploration of other metadata classifiers is also possible. We sort the predictions by confidence (independently for each measure of prediction confidence) and measure Acc@161 among the top-n% of predictions for the following values of n: $\{0.0, 0.05, ..., 1.0\}$, akin to a precision–recall curve, as shown in Figure 4.7 and Figure 4.8.[45] Results on Acc show a very similar trend, and are omitted.

The naive AP method is least reliable with, surprisingly, accuracy increasing as AP decreases in both figures. It appears that the raw probabilities are not an accurate reflection of prediction confidence. We find this is because a larger AP usually indicates a user has few LIW features, and the model often geolocates the user to

[45]Because we only evaluate on a subset of predictions, Acc@161 is equivalent to Precision@161. We decided to maintain consistent terminology, as in Cheng *et al.* (2010), and to use Acc@161.

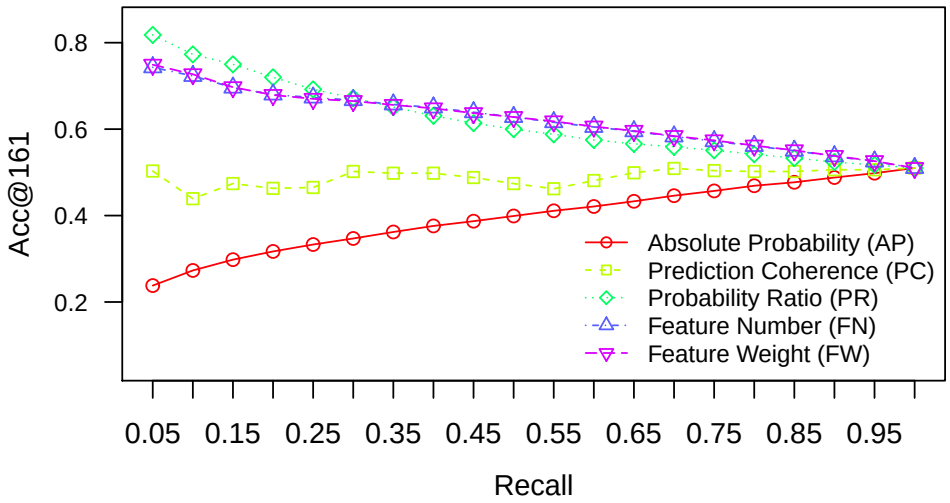


Figure 4.8: Acc@161 for classification of the top-n% most-confident predictions for each measure of text-based prediction confidence on LIVE.

the city with the highest class prior. In comparison, PR — which focuses on relative, as opposed to raw, probabilities — performs much better, with higher PR generally corresponding to higher accuracy. In addition, PC shows different trends on the two figures. It achieves comparable performance with PR on NA, however it is incapable of estimating the global prediction confidence. This is largely because world-level PC numbers are often very small and less discriminating than the regional PC numbers, reducing the utility of the geographic proximity of the top predictions. Furthermore, FN and FW display similar overall trends to PR, but don't outperform PR.

These experiments suggest that there is indeed a trade-off between coverage and accuracy, which could be further exploited to obtain higher-accuracy predictions in applications that do not require all the data to be classified. PR, as well as FN and FW, are fairly effective indicators of predictive accuracy. A further extension on this line of research would be to investigate the prediction confidence per city, e.g., are users from New York, US more predictable than users from Boston, US?

4.12 Summary

In this chapter, we have investigated a series of key issues relating to text-based geolocation prediction for Twitter users. We applied a number of feature selection methods to identify location indicative words (LIWs), and demonstrated the effectiveness of feature selection on improving the accuracy of geolocation prediction using different classifiers, location partitions, and datasets. Three feature selection methods (i.e., *IGR*, *Ripley*, *GeoDen*) achieve comparable best performance on two public datasets. In particular, a multinomial naive Bayes classifier using *IGR* features on the city-based partition outperforms the previous state-of-the-art benchmark geolocation prediction methods by 10.6 percentage points in terms of Acc@161, and reduces the median prediction error distance by 209km on NA dataset (Roller *et al.* 2012).

We then extended our study to analyse the impact of non-geotagged data, the influence of language and the complementary geographical information in the user metadata. We showed that by exploiting a user's non-geotagged tweets, the city-level accuracy is improved from 12.6% to 28.0% on WORLD, underlining the contribution of abundant non-geotagged data. Furthermore, the results also suggest that a model trained on geotagged data indeed generalises to non-geotagged data, although subdomain differences between geotagged data and non-geotagged data are observed. The results of language influence indicate that our model indeed generalises from a monolingual English to a multilingual setting. Furthermore, the experiments reveal that geolocation prediction is much easier for languages with more geographically-restricted use (e.g., Indonesian) than languages that are more diverse in usage (e.g., English). The best prediction result is obtained when training a number of monolingual geolocation models based on language identification and predicting locations using the same monolingual model with the user's primary language. As for the user metadata data, we found they also contains varying amount of geospatial information. In particular, modelling and inferencing on the basis of user-declared locations achieves the best single classifier accuracy, despite many declared locations being ad hoc and unstructured. The user metadata offers complementary information which can be further combined with that in tweet text. By incorporating information from

metadata and the tweet message in a stacking-based approach, we showed that a city-level accuracy of 49.1%, and a median prediction error distance of just 9km, can be achieved over our global dataset, which is a substantial improvement over any of the base classifiers.

We further evaluated our model on a time-heterogeneous dataset to assess the model's sensitivity to temporal change. The observed moderate decline in results indicates that the stacked geolocation model is indeed influenced by temporal change. Among the geospatial information sources, user-declared location is more sensitive to temporal change than tweet text and timezone information in our stacking model. Nonetheless, 40.6% city-level accuracy can still be obtained on the time-heterogeneous dataset, indicating the model generalisation.

Moreover, we discussed how a user's tweeting behaviour affects geolocation prediction and drew conclusions on how a user can make themselves less easily geolocatable. Experiments suggest the number of LIWs (in particular, gazetted terms) and user-declared metadata are key to geolocating a user, in addition to geotagged tweets.

Finally, we explored various indicators to estimate prediction confidence, in terms of the balance between prediction coverage and accuracy. The probability ratio, which measures the ratio of the probability of the top prediction with that of the second prediction, calibrates the prediction accuracy. The tweet text-based method can reach up to 80% accuracy for a selection of users, which is useful for applications that require higher accuracy, but are less demanding in recall.

To summarise, we improved the performance text-based geolocation prediction and examined many related influential factors. We believe these findings contribute to a deeper understanding of text-based geolocation prediction, and further shape the design of practical solutions to the problem.

Chapter 5

Conclusion

This thesis aims to improve the utility of social media data with NLP. In this chapter, we sum up our findings and contributions regarding this general theme. First, we briefly highlight the challenging issues in utilising social media data. Then, we present the research outcomes on two Twitter text processing tasks: text normalisation and geolocation prediction, addressed in Chapter 3 and Chapter 4, respectively. After that, we summarise the impact of the proposed tasks with respect to the research theme. Finally, we discuss the limitations and make suggestions for future work.

5.1 Summary of Findings

Social media sites (e.g., Twitter) generate massive volumes of user-generated text data which is often noisy, creating challenges for existing NLP tools and applications. Accuracy declines have been observed in various NLP tasks, such as POS tagging (Gimpel *et al.* 2011). Furthermore, the massive amount of text data (e.g., 500 million tweets per day) is potentially beyond the processing reach of existing tools. In addition, many applications like event detection require sufficient and reliable geospatial information which is often hard to obtain. There is a plethora of other data properties that hinder the utilisation of social media data such as ungrammatical sentence structure (Baldwin *et al.* 2013) and short document length.

To bridge the gap between noisy, large-scale social media data and existing NLP

tools, this thesis examined the tasks of text normalisation and geolocation prediction to improve the effectiveness and efficiency of social data utilisation with natural language processing. The findings are drawn from Twitter text processing. Nonetheless, the developed methods can be generalised to many other types of social media data, e.g., Facebook updates. The following is a summary of each task:

Text Normalisation: Non-standard words like *tmrw* and *4eva* in social media are generally not recognised by existing NLP tools, which are usually trained on edited text sources, e.g., newswire data. Text normalisation strives to transform these lexical variants (i.e., OOV non-standard words) to their canonical forms, to make them more amenable for downstream processing.

First, we conducted a pilot study on the coarse-grained categorisation of lexical variants, and found that many lexical variants are attributable to morphophonemic variations. This finding motivated us to pursue a token-based lexical normalisation method using a candidate generation-and-selection procedure. First, normalisation candidates are generated based on morphophonemic features of the OOV word. The number of generated candidates is balanced between efficiency and recall (i.e., the ratio of true normalisation among the generated candidates). Then, the most plausible candidate is selected for normalisation by combining various string and context similarities. We also compared the token-based method with a number of benchmarks, e.g., statistical machine translation and spell checking. The benchmark methods are not competitive for the task, either due to a lack of appropriate training data or because they are incapable of dealing with the range of lexical variants found in social media text. The unsatisfying performance of existing methods indicates that text normalisation is a challenging task.

By comparing and analysing the strengths and weaknesses of existing methods, we found that existing Internet slang lexicons cover many lexical variants and have high precision. Furthermore, we observed that the normalised forms of most longer lexical variants is often deterministic, e.g., while *hw* may be interpreted as "how" or "homework" depending on context, *tmrw* is usually normalised to "tomorrow". Inspired by these findings, we turned to a pure type-based approach which leverages distributional similarity and string similarity to derive a normalisation lexicon that directly

maps lexical variants to their standard forms. To generate the lexicon, we first extracted the most contextually similar IV words for an OOV, e.g., (*Mnday*, *Tuesday*), (*4eva*, *forever*), and (*Obama*, *Adam*). Then, we re-ranked these pairs by various string similarity methods to filter out undesired pairs that are used in similar contexts but are not the normalised forms of lexical variants, e.g., (*Mnday*, *Tuesday*) and (*Obama*, *Adam*). The ranked pairs are further pruned based on a small amount of annotated development data obtained through crowdsourcing. The resultant pairs then form a normalisation lexicon, which we found to be highly complementary to existing Internet slang dictionaries for normalisation.

Unlike the original token-based method whose overall effectiveness was also influenced by the unreliable detection of lexical variants, the lexicon-based method is an end-to-end solution, including both the detection and the normalisation of lexical variants. Combined with existing lexicons, the type-based approach achieved the best published lexical normalisation results at the time. Furthermore, the practical advantages of a pure lexicon-based approach are also clear: fast, lightweight and easily integrable into other components of a Twitter text processing pipeline.

The utility of text normalisation is further demonstrated by extrinsic evaluation on downstream tasks, e.g., our experiments on POS tagging and a number of tasks conducted by other researchers. To show the generality of our proposed pure lexicon-based approach, we also adapted the method to Spanish text normalisation, and achieved encouraging results. The concrete research outcomes are as follows:

- We simplified and formulated a lexical text normalisation task as a mapping from lexical variants (i.e., OOV non-standard words) to standard IV words relative to a dictionary. Based on morphophonemic clues, we proposed a token-based normalisation method using a candidate generation-and-selection approach that outperforms existing benchmarks.

- Comparisons of existing methods suggest text normalisation is a challenging task. Although the token-based approach attains better performance, it comes at the price of complex computation using various similarity calculations and an unrealistic assumption, i.e., perfect detection of lexical variants. Our pilot

study suggests the detection of lexical variants from standard OOV words (e.g., *Obama*) is also challenging, which further decreases the value of many existing methods. In contrast, we found the results of dictionary lookup relative to an Internet slang lexicon to be promising, with high precision and reasonable recall. This shifted the focus of text normalisation to a more practical and precision-oriented solution.

- Inspired by the benchmark results and analysis, we developed a pure type-based normalisation approach that maps longer unambiguous lexical variants to their standard forms. This type-based approach was the best published end-to-end normalisation solution at the time. The off-the-shelf combined lexicon achieves 0.847 precision and 0.630 recall on the benchmark dataset. It certainly has flaws in dealing with shorter and ambiguous lexical variants, however, it is also very fast, which is suitable for dealing with social media data.

- We further demonstrated the effectiveness of a lexicon-based approach in the context of a downstream Twitter POS tagging task. The comparison between text normalisation and an in-domain Twitter POS tagger suggests a trade-off strategy. As an off-the-shelf solution, lexicon-based method is a fast and universal text normalisation approach, and improves downstream results of out-of-domain tools over un-normalised data. As a comparison, an in-domain tool often has much higher accuracy, but requires non-trivial effort in data annotation and feature engineering.

- We adapted our type-based approach to Spanish text normalisation and achieved encouraging results. The automatically-derived lexicon is also highly complementary to the existing lexicons. This indicates the generality of the lexicon-based approach.

Geolocation Prediction: Geospatial information is useful in social media data partitioning and many downstream applications, e.g., local event detection. However, sufficient and reliable geospatial information is often not obtainable in social media. This task examined various ways to predict the geolocation of Twitter users using

only text data. Because word choice varies from region to region, our approach builds location "profiles" for each pre-defined location that consists of words that best distinguish that location from others. The basic approach is then to predict the user's location by matching their profile to the most similar "profile".

Words carry varying amounts of geospatial information. Location indicative words (e.g., gazetted terms) readily co-occur with common words without any geospatial dimension (e.g., stop words). The performance of text-based geolocation is often thwarted by these common words. To tackle this challenge, we improved the accuracy of geolocation prediction by automatically selecting location indicative words for location modelling and inference. We experimented with an extensive range of feature selection methods, and found *Ripley*, *IGR* and *GeoDen* to achieve the best performance on two different datasets. These methods are based on different heuristics or principles. For instance, *GeoDen* incorporates geometric proximity information. The geolocation prediction model trained on the resultant feature set outperformed the model trained on the full feature set.

In addition to investigating the impact of selecting and applying location indicative words in geolocation prediction, we also examined a number of other influential variables in a unified geolocation prediction framework. We found the choice of classification model and data partitioning to have a minor influence on geolocation prediction accuracy relative to the influence of feature selection (i.e., the identification of location indicative words). Geolocation prediction accuracy increases when non-geotagged tweets from training users are incorporated. The tweeting language also helps to narrow down the potential location. Due to the uneven geographical distribution of languages in tweets, users of geographically-diverse languages (e.g., English and Spanish) are much harder to geolocate than users of geographically-focused languages (e.g., Japanese or Dutch).

Instead of solely relying on tweet text data, many other sources in tweet metadata can also be integrated. In particular, we examined the use of user-declared location, timezone, user-declared description, and user registered real names. All these text sources offer some degree of geospatial information. Among them, we found the user-declared location to achieve the best prediction accuracy, despite many location

descriptions being ad hoc and unstructured. As a natural extension, we also experimented with combining these fields in stacking, and achieved substantially improved accuracy over any classifier using only one text source.

Additionally, we also examined the impact of temporal change on the trained geolocation model. The results suggest that accuracy decreases over time when training with data from a given time period. By further breaking down the stacked classifier into base classifiers, we found the accuracy decline is largely due to unreliable user-declared location data. In contrast, tweet text and timezone information are less prone to temporal change.

Having explored various factors to improve the geolocation prediction accuracy, we also studied how user tweeting behaviour affects geolocatability. We conducted ablative experiments to examine the contribution of each text source. The results provide insights for users to preserve privacy while using social media. Because many users don't mention geospatially related words, we investigated a number of variables that calibrate the predictions. We found that using the probability ratio — the ratio for the best prediction score over the second best prediction score — helps in distinguishing reliable predictions from less confident predictions.

A number of conclusions can be drawn from this study, corresponding to different sections in Chapter 4. We believe these findings contribute to a deeper understanding of text-based geolocation prediction:

- We demonstrated that explicit selection of location indicative words improves geolocation prediction accuracy, as compared to using the full feature set. A multinomial Bayes classifier on a city-based partition improves the previous state-of-the-art system by 10.6% in Acc@161 and 209km in median error distance. In addition, using only location indicative words as features also leads to compact models and faster prediction speed.

- Non-geotagged tweets (from users whose location is known) boost the prediction accuracy substantially in both training and testing. This is largely because of the similarity between geotagged data and non-geotagged data, although minor differences are observed between geotagged and non-geotagged tweets. We also

found that modelling on geotagged data and inferencing on non-geotagged data is indeed feasible.

- Modelling and inference on multilingual data is viable and easier than on monolingual English data. By integrating language information in different ways, we found training a range of monolingual models based on language identification, and predicting location using a model based on the user's primary language, achieves better results than a monolithic multilingual model.

- User-declared metadata, though noisy and unstructured, offers complementary geospatial information to what is contained in tweets. Specifically, simple statistical modelling based on user-declared location fields achieves 40.5% accuracy on our time-homogeneous test data, better than inferencing on tweet text data. By combining tweet and metadata information through stacking, the best global geolocation results are attained: over 49% of English users can be correctly predicted at the city level, with a median error distance of just 9km.

- Results on time-heterogeneous data suggest applying a model trained on "old" data to predict "new" data is generally feasible. Although the user-declared location field is sensitive to temporal change, classifiers based on the tweet text and user timezone generalise reasonably well across time.

- Our pilot study on user geolocatability led to the following recommendations to preserve geolocation privacy: (1) reduce the usage of location indicative words, particularly gazetted terms; and (2) delete location-sensitive metadata (e.g., user-declared location).

- Probability ratio, which measures the ratio of the probability of the top prediction with that of the second prediction, can be used to estimate prediction confidence, and select only users where the system prediction is more accurate, e.g., for downstream applications that require more-reliable geolocation predictions and where exhaustive user geolocation is not required.

Among various possible tasks relating to social media text processing, this thesis identified and closely examined text normalisation and geolocation prediction in Twitter text. Based on existing approaches, we investigated various ways to improve the accuracy on these two tasks. We not only improved the accuracy of two Twitter processing tasks, but also delivered practical methods which are suitable for processing large-scale social media data. For example, our type-based lexical normalisation approach is a fast and easy-to-use end-to-end solution, and our models for geolocation prediction are based on linear classifiers which easily scale up to larger datasets.

Due to the limited availability of datasets and off-the-shelf systems, the two tasks were largely evaluated using intrinsic metrics such as accuracy and F-score. As discussed in Chapter 1 and Chapter 2, improvements in text normalisation and geolocation prediction can potentially benefit many downstream NLP tasks and applications. A number of these tasks and applications are summarised and discussed in the thesis. For instance, text normalisation improves POS tagging (in Section 3.4) and machine translation (in Section 3.6). The predicted geospatial information enables location-based data partitioning for applications such as local event detection and regional sentiment analysis, as discussed in Sections 1.2 and 2.3.

To summarise, text normalisation and geolocation prediction improve the effectiveness and efficiency of social media utilisation, and advance the reach of social media text processing.

5.2 Limitations and Future Work

In this section, we identify a number of limitations and potential improvements to text normalisation and geolocation prediction. We also expand the discussion to consider the interaction of the two tasks.

Text Normalisation: The lexicon-based method can be further improved in several directions. Although the automatically-derived lexicon is highly complementary to existing lexicons, the quality of the derived lexicon is not comparably high. Currently, lexicon entries are firstly generated by distributional similarity and then filtered by string similarity. In this setting, correct normalisations might be skipped

due to the limitation of our distributional similarity implementation. For instance, the optimal size of context window in calculating distributional similarity differs from word to word, and a unified window size (i.e., ±2 two words around the OOV optimised on development data) may not be suitable for all words. As a result, the best normalisation candidate for a given OOV word may not necessarily be the most distributionally-similar IV word. Taking *Mnday* in our lexicon for example, its most distributionally-similar IV word is not "Monday" but instead "Tuesday", and consequently the (*Mnday*, *Monday*) entry won't be paired after the re-ranking step. To improve the lexicon quality, we intend to explore alternative ways to combine distributional and string similarity

One potential way is to allow more distributionally-similar normalisation candidates in the re-ranking step. For instance, "Monday" and "Tuesday" are both top ranked distributionally-similar candidates for *Mnday*. Instead of just using the most distributionally-similar word "Tuesday", the top-n candidates (including, for example, "Monday" and "Wednesday") should also be pushed into the string similarity re-ranking to select the most plausible standard form for *Mnday*. Because "Monday" is more similar to *Mnday* in terms of surface form, it may potentially rank higher than other normalisation candidates, leading to the correct normalisation (*Mnday*, *Monday*). The selection of candidate number (i.e., n) in the re-ranking step can be optimised using a small amount of development data.

Additionally, we have compared different string similarity methods for their effectiveness in ranking candidate normalisation pairs. These methods measured different aspects of string similarity such as character and phonetic variation. We could potentially improve the lexicon quality by combining these methods together in classifiers. For instance, the outputs of the string similarity methods can be vectorised as $\mathbf{x}$. A feature weight is attached to each method, forming the corresponding weight vector $\mathbf{w}$. A classifier f then takes $\mathbf{w}$ and $\mathbf{x}$ to generate a real-valued output $y = f(\mathbf{w} \cdot \mathbf{x})$, which is used to determine whether the normalisation entry is correct. The choice of the classifier is flexible, ranging from simple logistic regression to advanced support vector machines (SVM). The downside of this combination is that the optimisation of $\mathbf{w}$ involves supervised learning with training data. However, given that classifiers like

SVM often generalise well on a small amount of training data, this weakness doesn't seem to be a major impediment. A small amount of training data could be manually developed or crowdsourced as in Section 3.3.3. Furthermore, SVMs have the property of being less prone to feature redundancy. Because string similarity methods are often correlated to some degree, such as edit distance and consonant edit distance, this property of SVMs particularly suits our task.

Ultimately, context-sensitive normalisation is much more powerful than a lexicon-based approach, and we also plan to explore context-sensitive methods for token-based normalisation. Recent work on sequential labelling (Hassan and Menezes 2013) collectively determines the optimal normalisation for an entire tweet. The top-n normalisation candidates for each OOV are calculated, and the candidates are jointly selected so that the normalised tweet probability can be maximised relative to a language model.

In addition to such advanced models, a lightweight context-sensitive token-based normalisation can also be developed from the normalisation lexicon. For instance, *hw* would be normalised to "homework" in Examples (5.1)–(5.2), because it has words like *teacher* and *pages* in its context of use. In contrast, *hw* in Examples (5.3)–(5.4) would be normalised to "how", because the context contains words like *Hey* and *about*.

(5.1) I can't start my hw till my teacher emails back soooo I guess I'll just go eat

(5.2) When ur teacher assigns 20 pages of hw on the first night.

(5.3) Hey fwends! Hw r u guys doing

(5.4) "The boys are opening a coffee shop" Hw about no

Motivated by these examples, a lexicon-based approach can still be applied with lightweight context inference. For instance, we can identify the top-n normalisation candidates for each lexical variant in our lexicon based on distributional and string similarity. When encountering *hw* in an input tweet for example, we could calculate the similarity between *hw*'s tweet context (e.g., words near *hw*, which are often represented in a context word vector) and a candidate's context. The candidate with the

highest similarity could then be selected as the normalisation for *hw* in that tweet. The context of each normalisation candidate could be modelled using large volumes of tweet data. To further improve the efficiency, context words for each candidate could be weighted using pointwise mutual information (PMI) or other lexical association measures, and pruned by keeping only the top ranked terms, e.g., if *teacher* co-occurred with *homework* more often than expected, it would be kept in the context vector. The construction of these candidate contexts could happen offline relative to a background corpus. Thanks to the data sparsity of context words, the calculations and candidate selection should be very fast in practise. Additionally, the resultant similarity value could be incorporated as an extra feature in the SVM classifier to further boost normalisation performance.

In this thesis, the IV lexicon is from `Aspell`. It is relatively small and inadequate at capturing new words and named entities, which weakens the power of normalisation. To this end, adopting a corpus-derived lexicon could be more appropriate, e.g., unique word types with occurrences larger than a pre-defined threshold in a corpus can be used as the IV lexicon. One possible choice is Wikipedia data which contains many named entities, is clean and is continuously updated.

The text normalisation task setting can be adapted to different applications. For instance, capitalisation information is vital to NER (Arnav Khare 2006). Twitter messages contain many incorrect capitalisations which was largely ignored in this thesis (except for the simple approach used in the context of Spanish text normalisation). Normalising words as well as their capitalisations has the potential to further improve the accuracy of NER. In addition, normalisation can be tailored to different social communities (O'Connor *et al.* 2011; Bergsma *et al.* 2013). While *yolo* "you only live once" is often understood by younger generations, it should be normalised to the full deabbreviated form for other users.

Finally, as a means of improving the utility of social media text data, the evaluation of text normalisation should be expanded to more downstream NLP tasks and applications. Due to the lack of evaluation datasets for Twitter, the impact of text normalisation has only been verified on a limited number of tasks at the moment, e.g., Twitter POS tagging and machine translation. Nevertheless, we plan to perform

extra evaluations on tasks such as dependency parsing and keyword-based event detection in the future, as other appropriate resources and tools become available to enable these evaluations.

Geolocation Prediction: The research on geolocation prediction could also be expanded in a number of directions. To select location indicative words (LIWs), we experimented with various feature selection methods on tweet text. The same treatment can be applied to the metadata fields as well, e.g., user-declared location and user descriptions. Furthermore, we plan to refine the feature set by only preserving LIWs that are less sensitive to temporal change. For instance, *Olympic* was an indicative word for London around 2010, but its geospatial focus will change four (or even two) years later. In contrast, *yinz* is a local word primarily used in Pittsburgh, and is less likely to change over time. As a result, *yinz* is superior to *Olympics* as a LIW over time. We can harvest geotagged tweets periodically and rank words relative to feature selection methods in each time period. If some words are constantly ranked highly over time, then we can reliably believe they are time-invariant LIWs.

In this thesis, only alphabetic word unigrams are used in training models and prediction. Twitter specific entities, such as hashtags and user mentions are excluded. This certainly makes our approach generally applicable to other text sources such as Facebook updates. Alternatively, we can treat Twitter entities as unigrams to attempt to further improve the geolocation prediction accuracy.

Most work (including this thesis) primarily utilises geotagged data, which accounts approximately 1% of all the Twitter data in training models and prediction. However, the ultimate goal is to infer locations for the majority of non-geotagged data. Based on our analysis in Section 4.6, there might be minor language differences between non-geotagged data and geotagged data from the same users, which may affect the geolocation model prediction accuracy. A larger gap may be observed between users who only have non-geotagged data and the geotagged users we used in our experiments. Priedhorsky *et al.* (2014) has shown the difference between geotagged tweets and non-geotagged tweets is minor by comparing the unigram frequency of the two sources. Nonetheless, their work is based on individual tweets, where we plan to perform the same experiments on users, i.e., unigram frequency comparison between

users with only non-geotagged data and users with geotagged data.

Hierarchical classification models (Mahmud *et al.* 2012; Ahmed *et al.* 2013) are becoming increasingly popular, and could be compared with our stacked classification model. Hierarchical models follow a coarse-to-fine grained disambiguation process. A set of locations — not necessarily geographically close to each other — are recursively selected as the potential predictions until one single location is chosen. The classification at each level will eliminate other sets of locations at the same hierarchical level. This will accelerate the prediction speed owing to fewer comparisons. However, the downside is if there were an early prediction error, the final prediction result would not be correct.

Although explicit social network data (e.g., followers) can be non-trivial to retrieve due to Twitter API rate limits, user interactions can be reconstructed from the content of tweets with minor effort, e.g., replies, retweets and user mentions (Jurgens 2013). This implicit network information can be used in a majority vote-based model, and the model can be further incorporated into the current stacking framework. Furthermore, the implicit network-based prediction can be combined with text-based geolocation methods to further improve the calibration of prediction accuracy, e.g., we could test whether predictions from the text-based and network-based methods that agree have a higher accuracy than when they disagree.

Having integrated various sources of information, it would also be interesting to know how well humans perform on the same location prediction task. An annotation task can be set up in Amazon Mechanical Turk, in which annotators are asked to identify a user's primary location based on the user's tweets. Additionally, our experiments are carried out at the user level, where it would be also interesting to know how different classification models perform at the tweet level.

Finally, although intrinsic evaluation of geolocation prediction is meaningful on its own, extrinsic evaluation is also helpful in demonstrating the effectiveness of automatically-inferred geospatial information. We are particularly interested in porting our geolocation prediction module to regional sentiment analysis and local event detection. The utility of geolocation prediction can be quantitatively evaluated by turning on and off the prediction in downstream task evaluation.

The interaction between geolocation prediction and text normalisation is also an interesting direction to explore. For instance, the selection of IV words in normalisation can be tailored to a user's location, which could be obtained through geolocation prediction. While *brekkie* is widely used to mean "breakfast" in Australia, it may not make sense for many Indian and Singapore English users who are not familiar with the word. As a result, *brekkie* should be normalised for Indian and Singapore users but left intact for Australian users. Furthermore, some non-standard words should be normalised to different standard forms relative to users' geographical locations. *USC* is such an example which can be normalised to "University of Southern California", "University of South Carolina" and a range of other names.

Conversely, normalisation may have a negative impact on geolocation prediction, because geospatial information is lost during the transformation (Owoputi *et al.* 2013). For instance, while *Philadelphia* may be mentioned in many places as the formal name for this city, *Kiladelphia* is largely used in Philadelphia as a vernacular name by locals. Such lexical variants could be utilised in geolocation prediction. As a result, geolocation prediction should be performed ahead of text normalisation.

5.3 A Tweet-length Summary of Thesis

Finally, we present a tweet-length summary on the thesis as a take-away message: *U wanna undrstnd TWEETS lik dis? Make them readable like this; and if there are too many to read, select those from a region.*

Bibliography

Abrol, Satyen, and Latifur Khan. 2010. Tweethood: Agglomerative clustering on fuzzy k-closest friends with variable depth for location mining. In *2010 IEEE Second International Conference on Privacy, Security, Risk and Trust, and 2011 IEEE Second International Conference on Social Computing (PASSAT/SocialCom)*, 153–160, Minneapolis, USA.

——, ——, and Bhavani M. Thuraisingham. 2012. Tweeque: Spatio-temporal analysis of social networks for location mining using graph partitioning. In *2012 International Conference on Social Informatics*, 145–148, Washington, D.C., USA.

Adams, Benjamin, and Krzysztof Janowicz. 2012. On the geo-indicativeness of non-georeferenced text. In *Proceedings of the 6th International Conference on Weblogs and Social Media (ICWSM 2012)*, 375–378, Dublin, Ireland.

Ahmad, Farooq, and Grzegorz Kondrak. 2005. Learning a spelling error model from search query logs. In *Proceedings of the 2005 Conference on Empirical Methods in Natural Language Processing (EMNLP 2005)*, 955–962, Vancouver, Canada.

Ahmed, Amr, Liangjie Hong, and Alexander J. Smola. 2013. Hierarchical geographical modeling of user locations from social media posts. In *Proceedings of the 22nd International Conference on the World Wide Web (WWW 2013)*, 25–36, Rio de Janeiro, Brazil.

Alegria, Iñaki, Nora Aranberri, Víctor Fresno, Pablo Gamallo, Lluís Padró, Iñaki San Vicente, Jordi Turmo, and Arkaitz Zubiaga. 2013. Tweet normalization workshop at SEPLN 2013: An overview. In *Proceedings of the Tweet Normalization Workshop at SEPLN 2013*, 1–9, Mardrid, Spain.

Alejandro Mosquera, Elena Lloret, and Paloma Moreda. 2012. Towards facilitating the accessibility of web 2.0 texts through text normalization resources. In *Proceedings of the LREC workshop: Natural Language Processing for Improving Textual Accessibility (NLP4ITA)*, 9–13, Istambul, Turkey.

Amitay, Einat, Nadav Har'El, Ron Sivan, and Aya Soffer. 2004. Web-a-where: geotagging web content. In *Proceedings of 27th International ACM-SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2004)*, 273–280, Sheffield, UK.

Arnav Khare, 2006. Joint learning for named entity recognition and capitalization generation. Master's thesis, University of Edinburgh.

Aw, AiTi, Min Zhang, Juan Xiao, and Jian Su. 2006. A phrase-based statistical model for SMS text normalization. In *Proceedings of COLING/ACL 2006*, 33–40, Sydney, Australia.

Backstrom, Lars, Jon Kleinberg, Ravi Kumar, and Jasmine Novak. 2008. Spatial variation in search engine queries. In *Proceedings of the 17th International Conference on the World Wide Web (WWW 2008)*, 357–366, Beijing, China.

——, Eric Sun, and Cameron Marlow. 2010. Find me if you can: improving geographical prediction with social and spatial proximity. In *Proceedings of the 19th International Conference on the World Wide Web (WWW 2010)*, 61–70, Raleigh, USA.

Baldwin, Timothy, Paul Cook, Marco Lui, Andrew MacKinlay, and Li Wang. 2013. How noisy social media text, how diffrnt social media sources? In *Proceedings of the 6th International Joint Conference on Natural Language Processing (IJCNLP 2013)*, 356–364, Nagoya, Japan.

——, and Marco Lui. 2010. Language identification: the long and the short of the matter. In *Proceedings of Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT 2010)*, 229–237, Los Angeles, USA.

Beaufort, Richard, Sophie Roekhaut, Louise-Amélie Cougnon, and Cédrick Fairon. 2010. A hybrid rule/model-based finite-state framework for normalizing SMS messages. In *Proceedings of the 48th Annual Meeting of the ACL (ACL 2010)*, 770–779, Uppsala, Sweden.

Bennett, Paul N., Filip Radlinski, Ryen W. White, and Emine Yilmaz. 2011. Inferring and using location metadata to personalize web search. In *Proceedings of 34st International ACM-SIGIR Conference on Research and Development in Information Retrieval (SIGIR'11)*, 135–144, Beijing, China.

Bentley, Jon Louis. 1975. Multidimensional binary search trees used for associative searching. *Communication of the ACM* 18.509–517.

Bergsma, Shane, Mark Dredze, Benjamin Van Durme, Theresa Wilson, and David Yarowsky. 2013. Broadly improving user classification via communication-based name and location clustering on Twitter. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT 2013)*, 1010–1019, Atlanta, USA.

Bilhaut, Frédérik, Thierry Charnois, Patrice Enjalbert, and Yann Mathet. 2003. Geographic reference analysis for geographic document querying. In *Proceedings of the HLT-NAACL 2003 Workshop on Analysis of Geographic References*, 55–62, Edmonton, Canada.

Blair, Charles R. 1960. A program for correcting spelling errors. *Information and Control* 3.60–67.

Blei, David M., Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet Allocation. *Journal of Machine Learning Research* 3.993–1022.

Blitzer, John, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing (EMNLP 2006)*, 120–128, Sydney, Australia.

Bollen, Johan, Huina Mao, and Xiaojun Zeng. 2011. Twitter mood predicts the stock market. *Journal of Computational Science* 2.1–8.

Brants, Thorsten, and Alex Franz, 2006. Web 1T 5-gram Version 1. `http://catalog.ldc.upenn.edu/LDC2006T13`.

Brill, Eric. 1992. A simple rule-based part of speech tagger. In *Proceedings of the Third Conference on Applied Natural Language Processing*, 152–155, Trento, Italy.

——, and Robert C. Moore. 2000. An improved error model for noisy channel spelling correction. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, 286–293, Hong Kong, China.

Brody, Samuel, and Nicholas Diakopoulos. 2011. Cooooooooooooooollllllllllllllll!!!!!!!!!!!!!!! using word lengthening to detect sentiment in microblogs. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*, 562–570, Edinburgh, UK.

Brown, Peter F., Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based n-gram models of natural language. *Computational Linguistics* 18.467–479.

——, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics* 19.263–311.

Burton, Kevin, Akshay Java, and Ian Soboroff. 2009. The ICWSM 2009 Spinn3r Dataset. In *Proceedings of the Third International Conference on Weblogs and Social Media (ICWSM 2009)*, San Jose, USA.

Buyukkokten, Orkut, Junghoo Cho, Hector Garcia-Molina, Luis Gravano, and Narayanan Shivakumar. 1999. Exploiting geographical location information of web pages. In *ACM SIGMOD Workshop on The Web and Databases (WebDB'99)*, 91–96, Philadelphia, USA.

Chandra, Swarup, Latifur Khan, and Fahad Bin Muhaya. 2011. Estimating Twitter user location using social interactions–A content based approach. In *2011 IEEE Third International Conference on Privacy, Security, Risk and Trust, and 2011 IEEE Third International Conference on Social Computing (PASSAT/SocialCom)*, 838–843, Boston, USA.

Chang, Hau-wen, Dongwon Lee, Eltaher M., and Jeongkyu Lee. 2012. @Phillies tweeting from Philly? Predicting Twitter user locations with spatial word usage. In *2012 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, 111–118, Istanbul, Turkey.

Cheng, Zhiyuan, James Caverlee, and Kyumin Lee. 2010. You are where you tweet: a content-based approach to geo-locating Twitter users. In *Proceedings of the 19th ACM Conference on Information and Knowledge Management (CIKM 2010)*, 759–768, Toronto, Canada.

Cho, Eunjoon, Seth A. Myers, and Jure Leskovec. 2011. Friendship and mobility: user movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2011)*, 1082–1090, San Diego, USA.

Choudhury, Monojit, Rahul Saraf, Vijit Jain, Animesh Mukherjee, Sudeshna Sarkar, and Anupam Basu. 2007. Investigation and modeling of the structure of texting language. *International Journal on Document Analysis and Recognition* 10.157–174.

Chrupała, Grzegorz. 2014. Normalizing tweets with edit scripts and recurrent neural embeddings. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL 2014)*, 680–686, Baltimore, USA.

Church, Kenneth Ward, and Patrick Hanks. 1989. Word association norms, mutual information, and lexicography. In *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics*, 76–83, Vancouver, Canada.

Contractor, Danish, Tanveer A. Faruquie, and L. Venkata Subramaniam. 2010. Unsupervised cleansing of noisy text. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010)*, 189–196, Beijing, China.

Cook, Paul, and Suzanne Stevenson. 2009. An unsupervised model for text message normalization. In *Proceedings of the Workshop on Computational Approaches to Linguistic Creativity (CALC '09)*, 71–78, Boulder, USA.

Crandall, David J., Lars Backstrom, Daniel Huttenlocher, and Jon Kleinberg. 2009. Mapping the world's photos. In *Proceedings of the 18th International Conference on the World Wide Web (WWW 2009)*, 761–770, Madrid, Spain.

Cucerzan, Silviu, and Eric Brill. 2004. Spelling correction as an iterative process that exploits the collective knowledge of web users. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing (EMNLP 2004)*, 293–300, Barcelona, Spain.

Dalvi, Nilesh, Ravi Kumar, and Bo Pang. 2012. Object matching in tweets with spatial models. In *Proceedings of the Fifth ACM International Conference on Web Search and Web Data Mining (WSDM 2012)*, 43–52, Seattle, USA.

Damerau, Fred J. 1964. A technique for computer detection and correction of spelling errors. *Communications of the ACM* 7.171–176.

Daumé, III, Hal, and Daniel Marcu. 2006. Domain adaptation for statistical classifiers. *Journal of Artificial Intelligence Research* 26.101–126.

David Graff, Christopher Cieri, 2003. English Gigaword. `http://catalog.ldc.upenn.edu/LDC2003T05`.

Davis Jr., Clodoveu A., Gisele L. Pappa, Diogo Rennó Rocha de Oliveira, and Filipe de L. Arcanjo. 2011. Inferring the location of Twitter messages based on user relationships. *Transactions in GIS* 15.735–751.

De Marneffe, Marie-Catherine, Bill Maccartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of the 5th International Conference on Language Resources and Evaluation (LREC 2006)*, 449–454, Genoa, Italy.

Derczynski, Leon, Diana Maynard, Niraj Aswani, and Kalina Bontcheva. 2013. Microblog-genre noise and impact on semantic annotation accuracy. In *Proceedings of the 24th ACM Conference on Hypertext and Social Media*, 21–30, Paris, France.

Ding, Junyan, Luis Gravano, and Narayanan Shivakumar. 2000. Computing geographical scopes of web resources. In *Proceedings of the 26th International Conference on Very Large Data Bases*, 545–556, Cairo, Egypt.

Dredze, Mark, Michael Paul, Shane Bergsma, and Hieu Tran. 2013. Carmen: A Twitter geolocation system with applications to public health. In *AAAI Workshop on Expanding the Boundaries of Health Informatics Using AI (HIAI)*, 20–24, Bellevue, USA.

Dunning, Ted. 1993. Accurate methods for the statistics of surprise and coincidence. *Computational Linguistics* 19.61–74.

Eisenstein, Jacob. 2013a. Phonological factors in social media writing. In *Proceedings of the Workshop on Language Analysis in Social Media*, 11–19, Atlanta, USA.

——. 2013b. What to do about bad language on the Internet. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT 2013)*, 359–369, Atlanta, USA.

——, Brendan O'Connor, Noah A. Smith, and Eric P. Xing. 2010. A latent variable model for geographic lexical variation. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing (EMNLP 2010)*, 1277–1287, Cambridge, USA.

Fan, Rong-En, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. LIBLINEAR: A library for large linear classification. *Journal of Machine Learning Research* 9.1871–1874.

Fink, Clayton, Christine D Piatko, James Mayfield, Tim Finin, and Justin Martineau. 2009. Geolocating blogs from their textual content. In *AAAI Spring Symposium: Social Semantic Web: Where Web 2.0 Meets Web 3.0*, 25–26, Palo Alto, USA.

Foster, Jennifer. 2010. "cba to check the spelling" investigating parser performance on discussion forum posts. In *Proceedings of Human Language Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT 2010)*, 381–384, Los Angeles, USA.

——, and Oistein Andersen. 2009. Generrate: Generating errors for use in grammatical error detection. In *Proceedings of the Fourth Workshop on Innovative Use of NLP for Building Educational Applications*, 82–90, Boulder, Colorado.

——, Özlem Çetinoglu, Joachim Wagner, Joseph L. Roux, Stephen Hogan, Joakim Nivre, Deirdre Hogan, and Josef van Genabith. 2011. #hardtoparse: POS tagging and parsing the Twitterverse. In *Proceedings of the AAAI-11 Workshop on Analyzing Microtext*, 20–25, San Francisco, USA.

Gao, Jianfeng, Xiaolong Li, Daniel Micol, Chris Quirk, and Xu Sun. 2010. A large scale ranker-based system for search query spelling correction. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010)*, 358–366, Beijing, China.

Gelernter, Judith, and Nikolai Mushegian. 2011. Geo-parsing messages from microtext. *Transactions in GIS* 15.753–773.

Gimpel, Kevin, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. 2011. Part-of-speech tagging for Twitter: Annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL HLT 2011)*, 42–47, Portland, USA.

Golding, Andrew R., and Dan Roth. 1999. A winnow-based approach to context-sensitive spelling correction. *Machine Learning* 34.107–130.

Gonzalez, Rodolfo, Gerardo Figueroa, and Yi-Shin Chen. 2012. Tweolocator: a non-intrusive geographical locator system for Twitter. In *Proceedings of the 5th International Workshop on Location-Based Social Networks*, 24–31, Redondo Beach, USA.

Goolsby, Rebecca. 2010. Social media as crisis platform: The future of community maps/crisis maps. *ACM Transactions on Intelligent Systems and Technology (TIST)* 1.7:1–7:11.

Gouws, Stephan, Dirk Hovy, and Donald Metzler. 2011a. Unsupervised mining of lexical variants from noisy text. In *Proceedings of the First Workshop on Unsupervised Learning in NLP*, 82–90, Edinburgh, UK.

——, Donald Metzler, Congxing Cai, and Eduard Hovy. 2011b. Contextual bearing on linguistic variation in social media. In *Proceedings of the Workshop on Language in Social Media (LSM 2011)*, 20–29, Portland, USA.

Graham, Mark, Scott A. Hale, and Devin Gaffney. 2013. Where in the world are you? Geolocation and language identification in Twitter. *The Professional Geographer* . To appear.

Gravano, Luis, Vasileios Hatzivassiloglou, and Richard Lichtenstein. 2003. Categorizing web queries according to geographical locality. In *Proceedings of the 12th ACM Conference on Information and Knowledge Management (CIKM 2003)*, 325–333, New Orleans, USA.

Grier, Chris, Kurt Thomas, Vern Paxson, and Michael Zhang. 2010. @spam: the underground on 140 characters or less. In *Proceedings of the 17th ACM Conference on Computer and Communications Security*, 27–37, Chicago, USA.

Gruzd, Anatoliy, Barry Wellman, and Yuri Takhteyev. 2011. Imagining Twitter as an Imagined Community. *American Behavioral Scientist* 55.1294–1318.

Guyon, Isabelle, and André Elisseeff. 2003. An introduction to variable and feature selection. *Journal of Machine Learning Research* 3.1157–1182.

Hassan, Hany, and Arul Menezes. 2013. Social text normalization using contextual graph random walks. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL 2013)*, 1577–1586, Sofia, Bulgaria.

Hauff, Claudia, and Geert-Jan Houben. 2012. Geo-location estimation of Flickr images: social web based enrichment. In *Proceedings of the 34th European conference on Advances in Information Retrieval*, 85–96, Barcelona, Spain.

Hecht, Brent, Lichan Hong, Bongwon Suh, and Ed H. Chi. 2011. Tweets from Justin Bieber's heart: the dynamics of the location field in user profiles. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 237–246, Vancouver, Canada.

Hiltz, Starr R., and Murray Turoff. 1985. Structuring computer-mediated communication systems to avoid information overload. *Communication of the ACM* 28.680–689.

Hirst, Graeme, and Alexander Budanitsky. 2005. Correcting real-word spelling errors by restoring lexical cohesion. *Natural Language Engineering* 11.87–111.

Hoffmann, Donna L., and Marek Fodor. 2010. Can you measure the ROI of your social media marketing? *MIT Sloan Management Review* 52.40–49.

Hong, Liangjie, Amr Ahmed, Siva Gurumurthy, Alexander J. Smola, and Kostas Tsioutsiouliklis. 2012. Discovering geographical topics in the Twitter stream. In *Proceedings of the 21st International Conference on the World Wide Web (WWW 2012)*, 769–778, Lyon, France.

Hong, Lichan, Gregorio Convertino, and Ed H. Chi. 2011. Language matters in Twitter a large scale study. In *Proceedings of the 5th International Conference on Weblogs and Social Media (ICWSM 2011)*, 518–521, Barcelona, Spain.

How, Yijue, and Min-Yen Kan. 2005. Optimizing predictive text entry for short message service on mobile phones. In *Human Computer Interfaces International (HCII 05)*, Las Vegas, USA.

Hu, Yuheng, Ajita John, Dorée Seligmann, and Fei Wang. 2012. What were the tweets about? Topical associations between public events and Twitter feeds. In *Proceedings of the 6th International Conference on Weblogs and Social Media (ICWSM 2012)*, 154–161, Dublin, Ireland.

——, Kartik Talamadupula, and Subbarao Kambhampati. 2013. Dude, srsly?: The surprisingly formal nature of Twitter's language. In *Proceedings of the 7th International Conference on Weblogs and Social Media (ICWSM 2013)*, 245—253, Boston, USA.

Ireson, Neil, and Fabio Ciravegna. 2010. Toponym resolution in social media. In *Proceedings of the 9th International Semantic Web Conference on The Semantic Web - Volume Part I*, 370–385, Shanghai, China.

Izumi, Emi, Kiyotaka Uchimoto, Toyomi Saiga, Thepchai Supnithi, and Hitoshi Isahara. 2003. Automatic error detection in the Japanese learners' English spoken data. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL 2003)*, 145–148, Sapporo, Japan.

Jabeen, Saima, Sajid Shah, and Asma Latif. 2013. Named entity recognition and normalization in tweets towards text summarization. In *Proceedings of 2013 Eighth International Conference on Digital Information Management (ICDIM)*, 223–227, Islamabad, Pakistan.

Järvelin, Kalervo, and Jaana Kekäläinen. 2002. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems* 20.422–446.

Jiang, Long, Mo Yu, Ming Zhou, Xiaohua Liu, and Tiejun Zhao. 2011. Target-dependent Twitter sentiment classification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL HLT 2011)*, 151–160, Portland, USA.

John, George H., Ron Kohavi, and Karl Pfleger. 1994. Irrelevant features and the subset selection problem. In *Proceedings of the 11th International Conference on Machine Learning*, 121–129, New Brunswick, USA.

Jones, Lucy. 2010. The changing face of spelling on the Internet. Technical report. `http://www.spellingsociety.org/media/spelling-on-the-internet.pdf`.

Jurgens, David. 2013. That's what friends are for: Inferring location in online social media platforms based on social relationships. In *Proceedings of the 7th International Conference on Weblogs and Social Media (ICWSM 2013)*, 273–282, Boston, USA.

Kaplan, Andreas M., and Michael Haenlein. 2010. Users of the world, unite! the challenges and opportunities of social media. *Business Horizons* 53.59–68.

Kaufmann, Joseph, and Jugal Kalita. 2010. Syntactic normalization of Twitter messages. In *International Conference on Natural Language Processing*, Kharagpur, India.

Kemighan, Mark D., Kenneth W. Church, and William A. Gale. 1990. A spelling correction program based on a noisy channel model. In *Proceedings of the 13th International Conference on Computational Linguistics (COLING '90)*, 205–210, Helsinki, Finland.

Kinsella, Sheila, Vanessa Murdock, and Neil O'Hare. 2011. "I'm eating a sandwich in Glasgow": Modeling locations with tweets. In *Proceedings of the 3rd International Workshop on Search and Mining User-Generated Contents*, 61–68, Glasgow, UK.

Klein, Dan, and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL 2003)*, 423–430, Sapporo, Japan.

Kobus, Catherine, François Yvon, and Géraldine Damnati. 2008. Normalizing SMS: are two metaphors better than one? In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008)*, 441–448, Manchester, UK.

Koehn, Philipp, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, Chris Dyer, Ondřej Bojar, Alexandra Constantin, and Evan Herbst. 2007. Moses: open source toolkit for statistical machine translation. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007)*, 177–180, Prague, Czech Republic.

——, Franz Josef Och, and Daniel Marcu. 2003. Statistical phrase-based translation. In *Proceedings of the 3rd International Conference on Human Language Technology Research and 4th Annual Meeting of the NAACL (HLT-NAACL 2003)*, 48–54, Edmonton, Canada.

Kohavi, Ron, and George H. John. 1997. Wrappers for feature subset selection. *Artificial Intelligence* 97.273–324.

Kukich, Karen. 1992. Techniques for automatically correcting words in text. *ACM Computing Surveys* 24.377–439.

Laere, Olivier Van, Jonathan Quinn, Steven Schockaert, and Bart Dhoedt. 2013a. Spatially-aware term selection for geotagging. *IEEE Transactions on Knowledge and Data Engineering* 99.221–234.

——, Steven Schockaert, and Bart Dhoedt. 2013b. Georeferencing Flickr resources based on textual meta-data. *Information Sciences* 238.52 – 74.

Lee, Lillian. 1999. Measures of distributional similarity. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, 25–32, College Park, USA.

Leidner, Jochen L., and Michael D. Lieberman. 2011. Detecting geographical references in the form of place names and associated spatial natural language. *SIGSPATIAL Special* 3.5–11.

Leidner, Jochen Lothar, 2007. *Toponym Resolution in Text: Annotation, Evaluation and Applications of Spatial Grounding of Place Names*. University of Edinburgh dissertation.

Li, Chen, and Yang Liu. 2012. Improving text normalization using character-blocks based models and system combination. In *Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012)*, 1587–1602, Mumbai, India.

Li, Mu, Yang Zhang, Muhua Zhu, and Ming Zhou. 2006. Exploring distributional similarity based models for query spelling correction. In *Proceedings of COLING/ACL 2006*, 1025–1032, Sydney, Australia.

Li, Rui, Shengjie Wang, and Kevin Chen-Chuan Chang. 2012a. Multiple location profiling for users and relationships from social network and content. *Proceedings of the VLDB Endowment* 5.1603–1614.

——, Shengjie Wang, Hongbo Deng, Rui Wang, and Kevin Chen-Chuan Chang. 2012b. Towards social user profiling: unified and discriminative influence model for inferring home locations. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2012)*, 1023–1031, Beijing, China.

Li, Wen, Pavel Serdyukov, Arjen P. de Vries, Carsten Eickhoff, and Martha Larson. 2011. The where in the tweet. In *Proceedings of the 20th ACM Conference on Information and Knowledge Management (CIKM 2011)*, 2473–2476, Glasgow, UK.

Lieberman, Michael D., and Jimmy Lin. 2009. You are where you edit: Locating wikipedia contributors through edit histories. In *Proceedings of the Third International Conference on Weblogs and Social Media (ICWSM 2009)*, 106–113, San Jose, USA.

Lin, Dekang. 1998. Automatic retrieval and clustering of similar words. In *Proceedings of the 36th Annual Meeting of the ACL and 17th International Conference on Computational Linguistics: COLING/ACL-98*, 768–774, Montreal, Canada.

Lin, Jianhua. 1991. Divergence measures based on the Shannon entropy. *IEEE Transactions on Information Theory* 37.145–151.

Ling, Wang, Chris Dyer, Alan W Black, and Isabel Trancoso. 2013. Paraphrasing 4 microblog normalization. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*, 73–84, Seattle, USA.

Liu, Fei, Fuliang Weng, and Xiao Jiang. 2012. A broad-coverage normalization system for social media language. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL 2012)*, 1035–1044, Jeju Island, Korea.

——, Fuliang Weng, Bingqing Wang, and Yang Liu. 2011a. Insertion, deletion, or substitution? Normalizing text messages without pre-categorization nor supervision. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL HLT 2011)*, 71–76, Portland, USA.

Liu, Xiaohua, Shaodian Zhang, Furu Wei, and Ming Zhou. 2011b. Recognizing named entities in tweets. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL HLT 2011)*, 359–367, Portland, USA.

Lodhi, Huma, Craig Saunders, John Shawe-Taylor, Nello Cristianini, and Chris Watkins. 2002. Text classification using string kernels. *Journal of Machine Learning Research* 2.419–444.

Lui, Marco, and Timothy Baldwin. 2011. Cross-domain feature selection for language identification. In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP 2011)*, 553–561, Chiang Mai, Thailand.

——, and ——. 2012. langid.py: An off-the-shelf language identification tool. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL 2012) Demo Session*, 25–30, Jeju Island, Korea.

Luo, Zhunchen, Miles Osborne, and Ting Wang. 2012. Opinion Retrieval in Twitter. In *Proceedings of the 6th International Conference on Weblogs and Social Media (ICWSM 2012)*, 507–510, Dublin, Ireland.

Mahmud, Jalal, Jeffrey Nichols, and Clemens Drews. 2012. Where is this tweet from? Inferring home locations of Twitter users. In *Proceedings of the 6th International Conference on Weblogs and Social Media (ICWSM 2012)*, 511–514, Dublin, Ireland.

Mao, Huina, Xin Shuai, and Apu Kapadia. 2011. Loose tweets: an analysis of privacy leaks on Twitter. In *Proceedings of the 10th Annual ACM Workshop on Privacy in the Electronic Society (WPES '11)*, 1–12, Chicago, USA.

Marcus, Mitchell P., Mary Ann Marcinkiewicz, and Beatrice Santorini. 1993. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics* 19.313–330.

Marton, Yuval, Nizar Habash, and Owen Rambow. 2010. Improving Arabic dependency parsing with lexical and inflectional morphological features. In *Proceedings of the NAACL HLT 2010 First Workshop on Statistical Parsing of Morphologically-Rich Languages*, 13–21, Los Angeles, USA.

Ng, Andrew Y., and Michael Jordan. 2002. On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes. In *Advances in Neural Information Processing Systems 14 (NIPS-02)*, 841–848, Vancouver, Canada.

Núnez-Redó, Manuela, Laura Díaz, José Gil, David González, and Joaquín Huerta. 2011. Discovery and integration of Web 2.0 content into geospatial information structures: a use case in wild fire monitoring. In *Proceedings of the 6th International Conference on Availability, Reliability and Security*, 50–68, Vienna, Austria.

O'Connor, Brendan, Jacob Eisenstein, Eric P. Xing, and Noah A. Smith. 2011. Discovering demographic language variation. In *Proceedings of the Workshop on Machine Learning for Social Computing*, Whistler, Canada.

——, Michel Krieger, and David Ahn. 2010. TweetMotif: Exploratory search and topic summarization for Twitter. In *Proceedings of the 4th International Conference on Weblogs and Social Media (ICWSM 2010)*, 384–385, Washington, USA.

O'Hare, Neil, and Vanessa Murdock. 2013. Modeling locations with social media. *Information Retrieval* 16.30–62.

O'Sullivan, David, and David J. Unwin. 2010. *Point Pattern Analysis*, 121–155. John Wiley & Sons, Inc.

Owoputi, Olutobi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A. Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT 2013)*, 380–390, Atlanta, USA.

Padró, Lluís, and Evgeny Stanilovsky. 2012. Freeling 3.0: Towards wider multilinguality. In *Proceedings of the 8th International Conference on Language Resources and Evaluation (LREC 2012)*, 2473–2479, Istanbul, Turkey.

Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the ACL and 3rd Annual Meeting of the NAACL (ACL-02)*, 311–318, Philadelphia, USA.

Paul, Michael J, and Mark Dredze. 2011. You are what you tweet: Analyzing Twitter for public health. In *Proceedings of the 5th International Conference on Weblogs and Social Media (ICWSM 2011)*, 265–272, Barcelona, Spain.

Pennell, Deana, and Yang Liu. 2011a. A character-level machine translation approach for normalization of SMS abbreviations. In *Proceedings of the 5th International Joint Conference on Natural Language Processing (IJCNLP 2011)*, 974–982, Chiang Mai, Thailand.

——, and ——. 2011b. Toward text message normalization: Modeling abbreviation generation. In *Proceedings of 2011 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '11)*, 5364–5367, Prague, Czech Republic.

Peterson, James L. 1980. Computer programs for detecting and correcting spelling errors. *Communications of the ACM* 23.676–687.

Petrov, Slav, and Dan Klein. 2008. Parsing German with latent variable grammars. In *Proceedings of the Workshop on Parsing German*, 33–39, Columbus, USA.

Petrović, Saša, Miles Osborne, and Victor Lavrenko. 2010. Streaming first story detection with application to Twitter. In *Proceedings of Human Language*

Technologies: The 11th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL HLT 2010), 181–189, Los Angeles, USA.

——, Miles Osborne, and Victor Lavrenko. 2012. Using paraphrases for improving first story detection in news and Twitter. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT 2012)*, 338–346, Montréal, Canada.

Philips, Lawrence. 2000. The Double Metaphone Search Algorithm. *C/C++ Users Journal* 18.38–43.

Pontes, Tatiana, Marisa Vasconcelos, Jussara Almeida, Ponnurangam Kumaraguru, and Virgilio Almeida. 2012. We know where you live: Privacy characterization of Foursquare behavior. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, 898–905, Pittsburgh, USA.

Porta, Jordi, and José-Luis Sancho. 2013. Word normalization in Twitter using finite-state transducers. In *Proceedings of the Tweet Normalization Workshop co-located with 29th Conference of the Spanish Society for Natural Language Processing (SEPLN 2013)*, volume 1086, 49–53, Madrid, Spain.

Priedhorsky, Reid, Aron Culotta, and Sara Y. Del Valle. 2014. Inferring the origin locations of tweets with quantitative confidence. In *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work and Social Computing*, Baltimore, USA. To appear.

Quercini, Gianluca, Hanan Samet, Jagan Sankaranarayanan, and Michael D. Lieberman. 2010. Determining the spatial reader scopes of news sources using local lexicons. In *Proceedings of the 18th SIGSPATIAL International Conference on Advances in Geographic Information Systems*, 43–52, San Jose, USA.

Quinlan, John Ross. 1993. *C4.5: Programs for Machine Learning*. San Mateo, USA: Morgan Kaufmann.

Rabiner, Lawrence R. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE* 77.257–286.

Ratnaparkhi, Adwait. 1996. A maximum entropy model for part-of-speech tagging. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-96)*, 133–142, Philadelphia, USA.

Ren, Kejiang, Shaowu Zhang, and Hongfei Lin. 2012. Where are you settling down: Geo-locating Twitter users based on tweets and social networks. In *Proceedings of the 8th Asia Information Retrieval Societies Conference (AIRS 2012)*, 150–161, Tianjin, China.

Ritter, Alan, Sam Clark, Mausam, and Oren Etzioni. 2011. Named entity recognition in tweets: An experimental study. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing (EMNLP 2011)*, 1524–1534, Edinburgh, UK.

Ritterman, Joshua, Miles Osborne, and Ewan Klein. 2009. Using prediction markets and Twitter to predict a swine flu pandemic. In *Proceedings of the 1st International Workshop on Mining Social Media (MSM 2009)*, Sevilla, Spain.

Roller, Stephen, Michael Speriosu, Sarat Rallapalli, Benjamin Wing, and Jason Baldridge. 2012. Supervised text-based geolocation using language models on an adaptive grid. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning 2012 (EMNLP-CoNLL 2012)*, 1500–1510, Jeju Island, Korea.

Rout, Dominic, Kalina Bontcheva, Daniel Preoţiuc-Pietro, and Trevor Cohn. 2013. Where's @wally?: A classification approach to geolocating users based on their social ties. In *Proceedings of the 24th ACM Conference on Hypertext and Social Media*, 11–20, Paris, France.

Sadilek, Adam, Henry Kautz, and Jeffrey P. Bigham. 2012a. Finding your friends and following them to where you are. In *Proceedings of the Fifth ACM International Conference on Web Search and Web Data Mining (WSDM 2012)*, 723–732, Seattle, USA.

——, Henry Kautz, and Vincent Silenzio. 2012b. Modeling spread of disease from social interactions. In *Proceedings of the 6th International Conference on Weblogs and Social Media (ICWSM 2012)*, 322–329, Dublin, Ireland.

Sage, Adam. 2013. The Facebook platform and the future of social research. In *Social Media, Sociality, and Survey Research*, 87–106. John Wiley & Sons, Inc.

Sakaki, Takeshi, Makoto Okazaki, and Yutaka Matsuo. 2010. Earthquake shakes Twitter users: real-time event detection by social sensors. In *Proceedings of the 19th International Conference on the World Wide Web (WWW 2010)*, 851–860, Raleigh, USA.

Schmid, Helmut. 1994. Probabilistic part-of-speech tagging using decision trees. In *International Conference on New Methods in Language Processing*, 44–49, Manchester, UK.

Schulz, Axel, Aristotelis Hadjakos, Heiko Paulheim, Johannes Nachtwey, and Max Mühlhäuser. 2013. A multi-indicator approach for geolocalization of tweets. In *Proceedings of the 7th International Conference on Weblogs and Social Media (ICWSM 2013)*, 573–582, Boston, USA.

Schwarm, S., and M. Ostendorf. 2002. Text normalization with varied data sources for conversational speech language modeling. In *Proceedings of 2002 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '02)*, 789–792, Orlando, USA.

Serdyukov, Pavel, Vanessa Murdock, and Roelof van Zwol. 2009. Placing Flickr photos on a map. In *Proceedings of 32nd International ACM-SIGIR Conference on Research and Development in Information Retrieval (SIGIR'09)*, 484–491, Boston, USA.

Shalev, Daphna, 2013. Text Normalization for First Story Detection in Twitter. Master's thesis, University of Edinburgh.

Silva, Mário J., Bruno Martins, Marcirio Silveira Chaves, Ana Paula Afonso, and Nuno Cardoso. 2006. Adding geographic scopes to web resources. *Computers, Environment and Urban Systems* 30.378–399.

Sproat, Richard, Alan W. Black, Stanley Chen, Shankar Kumar, Mari Ostendorf, and Christopher Richards. 2001. Normalization of non-standard words. *Computer Speech and Language* 15.287–333.

Stolcke, Andreas. 2002. SRILM - An Extensible Language Modeling Toolkit. In *Proceedings of the 7th International Conference on Spoken Language Processing*, 901–904, Denver, USA.

Sun, Guihua, Gao Cong, Xiaohua Liu, Chin-Yew Lin, and Ming Zhou. 2007. Mining sequential patterns and tree patterns to detect erroneous sentences. In *Proceedings of the 22nd Annual Conference on Artificial Intelligence (AAAI-07)*, 925–930, Vancouver, Canada.

Takahashi, H., N. Itoh, T. Amano, and A. Yamashita. 1990. A spelling correction method and its application to an OCR system. *Pattern Recognition* 23.363–377.

Takhteyev, Yuri, Anatoliy Gruzd, and Barry Wellman. 2012. Geography of Twitter networks. *Social Networks* 34.73–81.

Thurlow, Crispin, 2003. Generation txt? The sociolinguistics of young people's text-messaging. `http://extra.shu.ac.uk/daol/articles/v1/n1/a3/thurlow2002003-paper.html`.

Toutanova, Kristina, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 3rd International Conference on Human Language Technology Research and 4th Annual Meeting of the NAACL (HLT-NAACL 2003)*, 173–180, Edmonton, Canada.

——, and Robert C. Moore. 2002. Pronunciation modeling for improved spelling correction. In *Proceedings of the 40th Annual Meeting of the ACL and 3rd Annual Meeting of the NAACL (ACL-02)*, 144–151, Philadelphia, USA.

Tsur, Oren, and Ari Rappoport. 2012. What's in a hashtag?: Content based prediction of the spread of ideas in microblogging communities. In *Proceedings of the Fifth ACM International Conference on Web Search and Web Data Mining (WSDM 2012)*, 643–652, Seattle, USA.

Tuten, Tracy L. 2008. *Advertising 2.0: Social media marketing in a Web 2.0 world*. Westport, USA: Praeger Publishers.

Vieweg, Sarah, Amanda L. Hughes, Kate Starbird, and Leysia Palen. 2010. Microblogging during two natural hazards events: what Twitter may contribute to situational awareness. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, 1079–1088, Atlanta, USA.

Vincenty, Thaddeus. 1975. Direct and inverse solutions of geodesics on the ellipsoid with application of nested equations. *Survey Review* 22.88–93.

Wang, Chuang, Xing Xie, Lee Wang, Yansheng Lu, and Wei-Ying Ma. 2005a. Web resource geographic location classification and detection. In *Proceedings of the 14th International Conference on the World Wide Web (WWW 2005)*, 1138–1139, Chiba, Japan.

Wang, Lee, Chuang Wang, Xing Xie, Josh Forman, Yansheng Lu, Wei-Ying Ma, and Ying Li. 2005b. Detecting dominant locations from search queries. In *Proceedings of 28th International ACM-SIGIR Conference on Research and Development in Information Retrieval (SIGIR 2005)*, 424–431, Salvador, Brazil.

Wang, Pidong, and Hwee Tou Ng. 2013. A beam-search decoder for normalization of social media text with application to machine translation. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT 2013)*, 471–481, Atlanta, USA.

Weeds, Julie, David Weir, and Diana McCarthy. 2004. Characterising measures of lexical distributional similarity. In *Proceedings of the 20th International*

Conference on Computational Linguistics (COLING 2004), 1015–1021, Geneva, Switzerland.

Wellman, Barry. 1979. The community question: The intimate networks of East Yorkers. *American Journal of Sociology* 84.1201–1231.

——, Peter J. Carrington, and A. Hall. 1988. Networks as personal communities. In *Social structures: a network approach*, 130–184. Cambridge, UK: Cambridge University Press.

Wing, Benjamin P., and Jason Baldridge. 2011. Simple supervised document geolocation with geodesic grids. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL HLT 2011)*, 955–964, Portland, USA.

Wolpert, David H. 1992. Stacked generalization. *Neural Networks* 5.241–259.

Wong, Wilson, Wei Liu, and Mohammed Bennamoun. 2006. Integrated scoring for spelling error correction, abbreviation expansion and case restoration in dirty text. In *Proceedings of the Fifth Australasian Data Mining Conference (AusDM2006)*, 83–89, Sydney, Australia.

Xia, Yunqing, Kam-Fai Wong, and Wenjie Li. 2006. A phonetic-based approach to Chinese chat text normalization. In *Proceedings of COLING/ACL 2006*, 993–1000, Sydney, Australia.

Xu, Wei, Alan Ritter, and Ralph Grishman. 2013. Gathering and generating paraphrases from Twitter with application to normalization. In *Proceedings of the Sixth Workshop on Building and Using Comparable Corpora*, 121–128, Sofia, Bulgaria.

Xue, Zhenzhen, Dawei Yin, and Brian D. Davison. 2011. Normalizing microtext. In *Proceedings of the AAAI-11 Workshop on Analyzing Microtext*, 74–79, San Francisco, USA.

Yang, Yi, and Jacob Eisenstein. 2013. A log-linear model for unsupervised text normalization. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*, 61–72, Seattle, USA.

Yang, Yiming, and Jan O. Pedersen. 1997. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning*, 412–420, San Francisco, USA.

Yeh, Alexander. 2000. More accurate tests for the statistical significance of result differences. In *Proceedings of the 18th International Conference on Computational Linguistics (COLING 2000)*, 947–953, Saarbrücken, Germany.

Yi, Xing, Hema Raghavan, and Chris Leggetter. 2009. Discovering users' specific geo intention in web search. In *Proceedings of the 18th International Conference on the World Wide Web (WWW 2009)*, 481–490, Madrid, Spain.

Yin, Jie, A. Lampert, M. Cameron, B. Robinson, and R. Power. 2012. Using social media to enhance emergency situation awareness. *Intelligent Systems* 27.52–59.

Yin, Zhijun, Liangliang Cao, Jiawei Han, Chengxiang Zhai, and Thomas Huang. 2011. Geographical topic discovery and comparison. In *Proceedings of the 20th International Conference on the World Wide Web (WWW 2011)*, 247–256, Hyderabad, India.

Zhang, Congle, Tyler Baldwin, Howard Ho, Benny Kimelfeld, and Yunyao Li. 2013. Adaptive parser-centric text normalization. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (ACL 2013)*, 1159–1168, Sofia, Bulgaria.

Zhou, Zhi-Hua. 2012. *Ensemble Methods: Foundations and Algorithms*. Boca Raton, USA: Chapman & Hall/CRC.

Zhu, Conghui, Jie Tang, Hang Li, Hwee Tou Ng, and Tiejun Zhao. 2007. A unified tagging approach to text normalization. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics (ACL 2007)*, 688–695, Prague, Czech Republic.

Zobel, Justin, and Philip Dart. 1996. Phonetic string matching: lessons from information retrieval. In *Proceedings of 19th International ACM-SIGIR Conference on Research and Development in Information Retrieval (SIGIR'96)*, 166–172, Zurich, Switzerland.

Zong, Wenbo, Dan Wu, Aixin Sun, Ee-Peng Lim, and Dion Hoe-Lian Goh. 2005. On assigning place names to geography related web pages. In *ACM/IEEE Joint Conference on Digital Libraries*, 354–362, Denver, USA.

Appendix A

Appendix

The mapping between Penn and CMU POS tags from Section 3.4.

Penn POS Tags	CMU POS Tags
NN, NNS	N
PRP, WP	O
NNP, NNPS	^
MD, Tags startswith V	V
Tags startswith J	A
R, WRB	R
UH	!
WDT, DT, WP$, PRP$	D
IN, TO	P
CC	&
RP	T
EX, PDT	X
CD	$
FW, POS, SYM, LS	G
Non-alphabetic tags	,
Twitter specific tokens	Copy CMU POS tags
All other tags	Skipped in evaluation

Table A.1: The mapping between Penn and CMU POS tags